A Hybrid Control Multi-Agent System to Automate Fruit Harvest and Yield Management

Khalid Salah

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Mechanical Engineering

2019
UNIVERSITY OF CANTERBURY
Christchurch, New Zealand

Dedicated to:
The children of Yemen,
civil war victims,
and refugees.

Acknowledgements

My research journey took three years and three months, but it felt like a lifetime. I have learnt a lot and gained an appreciation for research and academic work. I would not have had completed my research and submitted this thesis if it was not for the people around me. I have amazing family and friends. I do not have the right words to describe my gratitude, but I will just do like every normal doctorate student and list their names. However, I just want to say that everyone around me is special, and we have had unforgettable memories together.

I will start by thanking my soulmate Ahlam Salah and my little angels Hanin and Jumanah. They are the most precious and patient family who coped with my research struggles. I want also to thank my lovely mother back home in Sana'a, Yemen Rahmah Ali and my siblings Maha, Basher, Mohammed, and Haifa. Even with the current situation in Yemen, they have never given up supporting me and being proud of my academic achievements. To the unforgettable person in my life my father Naji Salah who unfortunately is not around to read this, I would say *"You have taught me things about life which no educational system could have provided. You may rest in peace".*

I would like to thank my previous main supervisor XiaoQi Chen who was the only academic in the whole engineering department to reply to my email and supervised my research for three years. I also want to thank my current main supervisor Chris Pretty who came in the last three months, read every word in my thesis, and managed to boost my energy to submit in time. Especial thanks are also given to my co-supervisors Kourosh Neshatian and Varvara Vetrova.

Huge thanks to my awesome office mates and my friends Anthony FitzPatrick, Angus McGregor, and Arun Kumar who helped me, proofread my work, and provided informative technical discussions and solutions. We have had an amazing friendship. I want also to thank

the department of mechanical engineering staff and technicians for their cooperation and kind assistance. Lastly, I would like to thank the University of Canterbury scholarship department for funding this study.

ABSTRACT

Robotics and automation have significantly advanced modern industry in many aspects. In particular, automation has enabled companies to increase their outputs, improve the quality of finished work, and enhance production efficiency. Agriculture is one industry yet to be successfully automated, and is still highly dependent on labour. Currently, there is a rapid deterioration in the number of people working in the agricultural sector, at a time when growing global populations are causing an ever increasing demand for food. This is profoundly concerning, and clearly there is a need for research on agricultural automation.

Automating agricultural tasks has historically proven challenging, with complexities arising from dynamic outdoor environments, difficult terrains, and wide variety in plants morphologies. Moreover, previous studies on automating agricultural tasks have failed to deliver solutions broadly considered reliable by industry. Current research has indicated that full automation of agricultural activities is unrealistic. This thesis discusses how human intelligence and flexibility are still important factors which can be integrated with automation technology as part of a multi-agent system, and proposes the use of a Human – Robot Hybrid Control Multi-agent System to overcome outdoor automation challenges. A collaborative multi-agent model is a novel solution, and is developed here to reliably and optimally automate fruit harvest.

The Hybrid Control Multi-agent System aims to successfully automate fruit harvesting and yield management processes at a minimum economic cost. In this case, it is expected that a well-designed collaborative fruit harvesting system can reduce the yearly economic cost of a harvest by 60%. The hybrid control system's architecture is designed to be reliable by

evaluating the limitations of current and previous automation applications. The system architecture has a central controller to master plan the harvest process, while decentralized robotic transporting agents collaboratively serve fruit pickers and compensate for the absence of other robotic transporting agents and the central controller.

The proposed system architecture is modelled and simulated in realistic fruit harvesting scenarios to demonstrate its feasibility and behaviour. The model includes a virtual fruit orchard block represented by an undirected weighted graph; human picking agent models who pick fruit; mobile robot models who collect picked fruit from workers; and a tractor model which collects fruit bins located near workers. A Monte-Carlo simulation enhanced with a convergence analysis was used to ensure the reliability of results within the simulated framework. The model can be used for different kind of fruit, however, this study simulated apple fruit orchard. A first simulation was made by replacing the conventional setup of a human driven tractor and fixed picking bins, with a single tractor-equivalent robotic transporting agent which acts as a mobile fruit bin. The simulation found that the robotic transporting agent reduced a workers unproductive time by 41.3% when compared to a conventional apple harvesting setup. However, it was found that the single robot transporting agent resulted in high waiting times between a worker having a full bag and being served by the robot transporting agent. Replacing the tractor-equivalent robotic transporting agent with two smaller robotic transporting agents reduced the simulated service waiting time by 43.1%. These simulations both used a basic First Available First Served (FAFS) dispatching algorithm which assigns robotic transporting agents based on their availability.

To further reduce service waiting times and decrease RTAs utilisation, a Dynamic Distance dispatching algorithm (DD) replaced the FAFS algorithm to serve human picking agents based on both the availability and proximity of the robotic transporting agent to a service request. A simulation of 10 human picking agents and three small robotic transporting agents with the

new algorithm improved the mean service waiting time by 68.1% compared to the FAFS service algorithm. Further, the DD algorithm reduced robotic transporting agent utilisation from 45.7% to 15%. Serving capacity and the robotic transporting agents' utilisation are exponentially related, and the robotics transporting agents' utilisation should be below 80% to prevent human picking agents waiting to empty their bags for long time. A Dynamic Distance and Best fit (DDB) dispatching algorithm for heterogeneous robotic transporting agents which assigns them base on their availability, location, and speed reduced the mean of the service waiting time even further by 73.9% when compared to FAFS, and reduced the robotic transporting agents' utilisation to 12%. For the simulated orchard, an optimal cost is achieved when deploying three small, fast robotic transporting agents.

A smart picking bag system for fruit pickers has been prototyped, as a suitable commercial bag was not available. This technology is needed to test the system concept in a real orchard setting. The smart bag manages the fruit picking, tracks human picking agents, and allows for continuous human-machine interaction during the harvesting process. The smart bag measures the weight of fruit in the picking bag, and calls for a robotic transporting agent when required.

Technology was also developed to manage the robotic transporting agent's visual navigation through orchards from off-the-shelf and open source object detection algorithms. Initial testing of the visualisation technology successfully detected the human picking agents and calculated their distance. However, further testing and improvement of both technologies is required.

The proposed system is capable of reducing unproductive picking time by a significant fraction, and has the potential to reduce harvesting costs by approximately 60%. Work in developing the underlying dispatching algorithms has tripled the system's serving capacity and service time reliability from initial results. Initial prototypes of required technologies have been designed and field tested to manage the cooperative human-robot interaction, and robotic visual

navigation in a GPS denied environment. The proposed Hybrid Control Multi-agent System has clear benefits to the farming, and is an innovative and promising new attempt to automate the agricultural industry.

UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

Deputy Vice-Chancellor's Office
Postgraduate Research Office

Co-Authorship Form

This form is to accompany the submission of any thesis that contains research reported in co-authored work that has been published, accepted for publication, or submitted for publication. A copy of this form should be included for each co-authored work that is included in the thesis. Completed forms should be included at the front (after the thesis abstract) of each copy of the thesis submitted for examination and library deposit.

Please indicate the chapter/section/pages of this thesis that are extracted from co-authored work and provide details of the publication or submission from the extract comes:

1. Salah, K., X. Chen, K. Neshatian, V. Vetrova, and C. Pretty. *A Collaborative Hybrid Control Multi-Agent System to Automate Apple Harvest and Yield Management*. Submitted to IEEE Transactions on Automation Science and Engineering.
2. Salah, K., X. Chen, K. Neshatian, and C. Pretty. *A hybrid control multi-agent cooperative system for autonomous bin transport during apple harvest*. in *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 2018. (Published)
3. Salah, K., X. Chen, *Cooperative Robotic Systems in Agriculture*, in *Robotics and Mechatronics for Agriculture*, D. Zhang and B. Wei, Editors. 2017, CRC Press, Taylor & Francis Group. (Published)

Please detail the nature and extent (%) of contribution by the candidate:

I have contributed 90% of the published work.

Certification by Co-authors:

If there is more than one co-author then a single co-author can sign on behalf of all

The undersigned certifies that:

- The above statement correctly reflects the nature and extent of the PhD candidate's contribution to this co-authored work
- In cases where the candidate was the lead author of the co-authored work he or she wrote the text

Name: *Khalid Salah*	Signature: *Khalid Salah*	Date: *28/02/2019*

List of Publications

Salah, K., X. Chen, K. Neshatian, V. Vetrova, and C. Pretty. *A Collaborative Hybrid Control Multi-Agent System to Automate Apple Harvest and Yield Management*. Submitted to IEEE Transactions on Automation Science and Engineering.

Salah, K., X. Chen, K. Neshatian, and C. Pretty. *A hybrid control multi-agent cooperative system for autonomous bin transport during apple harvest*. in *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 2018. (Published)

Salah, K., X. Chen, *Cooperative Robotic Systems in Agriculture*, in *Robotics and Mechatronics for Agriculture*, D. Zhang and B. Wei, Editors. 2017, CRC Press, Taylor & Francis Group. (Published)

Table of Contents

List of figures

List of Tables

List of acronyms

AGV	Aerial Guided Vehicle
AFPM	Autonomous Fruit Picking Machine
BML	Battle Management Language
CLT	Central Limit Theorem
CMS	Central Processing Station
CSV	Comma-Separated value
CRIs	Crown Research Institutes
DFS	Depth-first search
DD	Dynamic Distance
DDB	Dynamic Distance and Best fit
FAFS	First Available First Serves
GIS	Geographic Information System
GPS	Global Positioning System
GHR	Grain Harvesting Robot
HPA	Human Picking Agent
HMI	Human Machine Interface
I^2C	Inter-integrated Circuit
MTBF	Mean Time Before Failure
MRTM	Multi-Robot Task Module
NZD	New Zealand Dollar
ORA	Ontology for Robotics and Automation
PC	Personal Computer
POS	Position Ontology
PA	Precision agriculture
PU	Primary Unit
RANSAC	Random Sample Consensus
RGB-D	Red, Green, Blue, and Depth
RTA	Robotic Transporting Agent
SPI	Serial Peripheral Interface
SSM	Site-Specific Management
SU	Supporting Unit
TFT LCD	Thin-Film-Transistor Liquid Crystal Display
UWB	Ultra Wideband
UAS	Unmanned Aircraft Systems
UGV	Unmanned Ground Vehicle

CHAPTER 1: INTRODUCTION

1.1 Introduction

This thesis addresses the gap between the automation technologies and how to successfully automate agricultural tasks. Automating agricultural tasks is crucial to maintain future food demand and overcome the agricultural labour shortage. However, automating most of the agricultural tasks have not been successful for the last 70 years for several reasons discussed in Chapter 2. Thus, this study proposed a collaborative multi-agent solution to automate harvest and yield management, which require intensive labour, by employing a cooperative behaviour between workers which overcomes agricultural environment complexity and automation limitations as shown in Fig. 1.1. The sizes of the bubbles in the Fig.1.1 represent the use of the technology which were estimated based on the author's knowledge of the field and previous studies. In other words, the sizes of the bubbles are not based on numerical data or quantitative analysis.

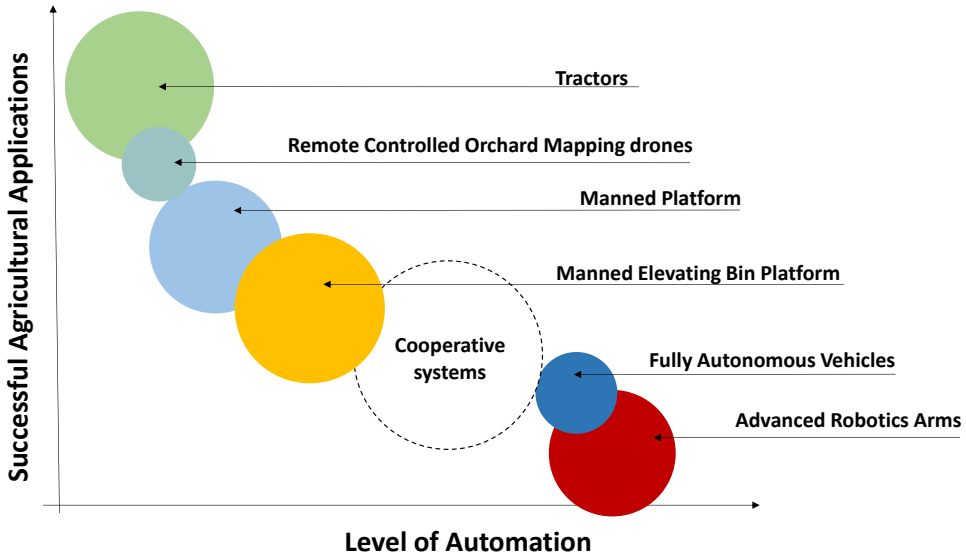


Fig. 1.1. The use of automation in agriculture and this study research focus.

The existing trend within primary production is to hire cheap overseas seasonal workers who are subjected to exploitation, egregious work conditions, and horrifying human rights abuses in New Zealand [1] and worldwide [2]. Another trend is to use bigger, heavier vehicles to perform agricultural operations in the shortest possible time causing soil damage and leading to extra soil treatment cost [3-5]. The reasons to adopt these approaches are the failure to automate basic agricultural tasks [6] and the advancement of industrial automation which enabled building cheap and massive agrarian vehicles [7].

Success of automation and robotics in production industries has created a cognitive bias that robots are the solution to replace labour and improve production [8]. This inaccurate judgement of technology made the engineering and research society try to develop agricultural robotic prototypes for the last century and none were successfully commercialized [6, 9]. Moreover, previous research has focused on adding advanced technology in the hope it would overcome poor performance of automation solutions and limitations in outdoor environments. An alternative solution to large and complex machine is to incorporate workers' flexibility and intelligence with basic mobile robots to collaboratively automat the harvest and yield management. The workers handle the intelligent works – picking fruit in a delicate way, while robots handle fruit dispatching and transporting.

1.2 Motivation and Objectives

The fruit industry is thriving in many countries. For example, NZ horticultural industry aims to extend its global position as a premium agricultural product supplier. The available land for horticultural farming is approximately 128000 ha which make up about 15% of NZ's farm land [10]. The apple industry contributed more than 24% of the total horticultural exports in 2013 [11]. Apple export income was 720 million New Zealand Dollar (NZD) in 2016. In order to achieve the NZ government target of 1 billion NZD apple export income by 2022 [12],

investment in new agricultural technologies and solutions will help increase the production at a lower cost.

Fruit farming and orchard management are expensive, complicated and conventional. It requires skilled labourers who work in hard conditions. The manual labour contributes to more than 60% of the total cost. The entire farming cost is made of planting, fertilizing, spraying and harvest management [13]. This thesis investigate a promising way of improving the fruit farming efficiency.

Harvest management is the most expensive procedure due to the need for seasonal workers during the fruit picking season, July-October in the northern hemisphere or February-May in the southern hemisphere. During the harvest, workers pick fruit, drive tractors and operate machinery. Currently, fruit bins placed in orchard rows are walked to and filled by pickers, inspected by supervisors, and transported by orchard tractors to a drop station. Such a harvest management procedure is common across the fruit industry costly and time-consuming.

Moreover, fruit picking is a combination of repetitive movements including picking, filling picking bags, and walking to fill fruit bins. Orchard workers are exposed to work stress, tripping hazards, and machinery related injuries. Besides, transporting bins by forklift tractors poses fatal injuries to workers and tractor drivers. Tractors are subjected to rollover risk which is the one of the main causes of agricultural work-related fatal accidents [14].

Completely automating the fruit harvesting process is an alternative solution, nevertheless, automating the whole process is difficult and requires complicated and robust systems due to orchards challenging structure and dynamic environment. Moreover, fruit have various shapes, variable sizes, delicate texture, and unreachable positions. Fruits are surrounded by branches and leaves which make it difficult to employ fast and flexible end effectors to automatically

pick them [15]. Thus, automating fruit picking is not viable. Another challenging task is to automate the yield management.

Small, lightweight, and simple mobile robots cooperate and serve a team of workers to automate the harvest process. The workers handle delicate tasks not suited to automation, while robots dispatch fruit from the workers and transport to collection stations. Such a collaborative robotic system achieves high productivity, lower cost and lower soil compaction making the production and operations more sustainable. Therefore, developing a collaborative robotic system is required to augment and automate tasks in agriculture sector without cutting out the human.

A collaborative multi-agent system is a practical solution to manage the interactions between pickers and fruit robots. The collaborative behaviour resolves fruit harvest automation complexity. A collaborative multi-agent system has a team of agents interacting with each other to complete a common task or set of functions [16]. An agent in precision agriculture is defined as an autonomous and computational system that can adapt to environmental changes [17]. This thesis investigates the design of a robust collaborative hybrid control multi-agent system to automate harvest and yield management in an unpredictable and dynamic outdoor environment. This system is made of an orchard, humans as fruit picking agents, decentralized controlled robotic transporting agents employed as mobile bins, and a central controller computer employed as a master planner.

The key challenges addressed in this thesis are:

- Design and develop a robust system to automate harvest and yield management.
- Determine the system's capacity and the optimal number of robotic transporting agents.
- Optimize the service waiting time.

- Design, develop, and test the human-robot interaction, human fruit picking agent tracking, and robotics navigation in orchard as a GPS denied environment.

1.3 Thesis Contribution

This thesis presents a means to efficiently automate harvest and yield management, optimize the harvest time and cost, and enhance workers productivity and safety. The cost is minimized by cutting out tractors, drivers, and supervisors since the supervisor role is to call a tractor to transport the yield and monitor the pickers. The system collaboratively manages human-robot interactions to resolve yield management complexity. The system contributes a novel yield management approach which incorporates humans' flexible manual skills with robots' precision and navigation ability. The system collects real-time data and provides performance analysis for future enhancement.

Chapters 3, 4, and 5 contributes a discrete-time event simulation model to simulate fruit harvest. The model has a virtual orchard model, multiple pickers with different experience model, a tractor model and multiple robotics model. The model enables simulating, developing dispatching algorithms, and testing different scenarios. The model provides a better understanding of the harvest process and results' visualization.

Chapter 6 contributes DD dispatching algorithm to optimize serving time and pickers' waiting time hence maximizes their productivity. It is designed to improve FAFS dispatching algorithm. The FAFS allows the central controller to assign the first available robot which resulting in long service waiting time. The DD allows the central controller to assign the most suitable robot transporting agents to dispatch pickers' bags based on their availability and location. The chapter also contributes another DDB dispatching algorithm to assign heterogeneous robotic transporting agents efficiently.

Chapter 7 and 8 contributes picking monitoring and visual robotics navigation system. The system is made of two subsystems which are smart picking bag carried for the pickers and a

visual robotics detection and navigation system. The system is design to enhance robotics navigation and workers detections in a GPS denied environment.

1.4 Thesis Structure

This thesis has eight chapters to describe the development of a reliable collaborative hybrid control multi-agent system to successfully automate harvest and yield management. The first chapter introduces the proposed system, motivation, research objectives, and expected outcomes. The remaining chapters are as follows:

Chapter 2. *Literature Review* presents a brief background of precision agriculture (PA) and an overview of the recent studies in the field of agricultural mechatronics and automation; collaborative robotic systems; and human-robot collaboration. The review which was published as a book chapter [18] focusing on the system architectures, collaborative behaviours and limitations.

Chapter 3. *Design of a Collaborative Hybrid Control Multi-Agent System* describes the system and its sub-systems architecture. The system consists of a centralised controller (Master Planner), a decentralised controller (robotic transporting agent) and smart picking bag to weigh the picked fruit and track the picker. The system automates fruit harvest and yield management by monitoring pickers and assigning autonomous mobile robots to dispatch fruit and transport to a collection station.

Chapter 4. *Modelling Fruit Harvest and Yield Management* presents real-life fruit orchards and harvest management practices, methods for modelling an orchard which is made of fruit tree rows, drive rows and a drop station. This chapter model yield dispatching and delivery. Apple orchard were used in the models as an example, which are based on real data obtained from a site visit to a commercial apple orchard and driven assumption for simplicity. Chapter 3 and 4 were published as a conference paper in ICIEA 2018, IEEE and was final listed within the best

6 papers [19]. It was extended to a journal paper and submitted to the industrial informatics IEEE transactions.

Chapter 5. *Simulating Apple Harvest, Current Yield Management Method and Automated Yield Management Method* presents the simulation of an apple harvest which simulates multiple human fruit picking agent (HPA) models with different experience, and robotic transporting agent (RTA) models to dispatch and deliver the yield. Different scenarios were simulated such as employing a single robotic transporting agent as an autonomous bin verses a tractor transporting fixed bins and employing of multiple robotic-agents.

Chapter 6. *Developing Despatching Algorithm to Optimize the Service Waiting Time and Number of Serving Robots, and soil damage* presents developing algorithm to assign available robots to dispatch the yield. The developed algorithm improve the service waiting time, number of serving robots, and soil compaction. The developed algorithms are First Available First Serves (FAFS) dispatching algorithm, Dynamic Distance (DD) dispatching algorithm for homogenous robots and Dynamic Distance and Best fit (DDB) dispatching algorithm for heterogeneous robots.

Chapter 7. *Developing a Fruit Picking Monitoring and Robotic Visual Navigation System in Orchards* presents the design and testing of system to monitoring picking process and enhance absolute robotics navigation in orchards as a GPS denied environment. The system is made of two sub-systems. The first sub-system is a smart fruit picking bag to actively monitor the pickers and enable human-robot interaction. The second sub-system is visual detection and navigation system to which enhances robotic navigation.

Chapter 8. *Conclusion and Future Development* presents a summary of the conducted research alongside with limitation and challenges. This chapter provides recommendations for future research.

1.5 Summary

The full automation of agricultural tasks has proven to be difficult and failed to be commercialized. The current commercial solutions to the labour shortage is to use bigger agrarian machines and import overseas labour during the harvest season. The need of labour during harvest is critical thus automating fruit harvest and yield management at a minimum cost while enhancing pickers' productivity and safety is the main focus of this thesis.

The main motivation of developing a multi-agent system to automate fruit harvest is the rapid growth of fruit industry and the crucial need for labour. Moreover, the agricultural industry has a labour shortage and costly. The proposed solution can be successful implemented by modelling and simulating fruit harvest.

Therefore, a discrete-event model is the base tool to analyse fruit harvest, fruit picking, yield dispatching, operation algorithms, and dispatching algorithms. The human-robot interaction is an essential part of the system which was achieved by designing a smart bag prototype. The visual navigation and human tracking system are developed to improve robotics navigation in orchards. The system is a PA product which can provide site-specific data about each picker, robot, tree, block, and orchard. The collected data provide fair payrolls and future labour enhancement. Besides automating harvest, the system can be utilized to automate precise spraying and fertilization.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

Automation is the long term solution to overcome labour shortage and optimize production. It had played a significant rule in manufacturing. Robotics and automation technology have enabled a nm precision products, for example, the semiconductor companies can produce a 5 nm node chips [20]. On the other hand, automation technology has been commercially used in meat processing and fruit sorting, grading, and packaging. However, it has had much success in automating agricultural and farming tasks [21]. This chapter reviews and discusses the development and limitation of automation and robotics applications in agriculture. It presents the concept of PA and the latest agricultural automation solutions including cooperative robotics systems and human to robot collaborative systems.

The increase in global population which is expected to reach 10 billion by 2050 has resulted in an increasing food demand and current agricultural practices will not be enough to meet future food demand [22]. As a consequence, primary industries must double agricultural production to meet such demand [22]. Increasing agricultural production cannot be simply achieved by doubling the resources such as water, lands, seeds, and chemicals. In fact, these resources have already been over-stretched causing environmental damage [22].

Enhancing the efficiency can optimize the production. The efficiency enhancement is achieved by improving the farming practices with the aid of technology [23]. Automation and robotics are good options to improve agricultural practices to achieve optimal and sustainable production [24, 25]. Moreover, automation and robotics technology can overcome the current agricultural labour shortage.

Agricultural automation solutions have been introduced and some technologies are commercially available. These solutions are manually operated, semi-automated and fully automated. However, some farming practices are difficult to be automated. Thus, new systems are required to aid the workers during plantation, harvesting, and supplying to the market [26]. For example, specialty crops farming is automation unfriendly which still dependent on labour [26].

In general, farming practices are complex to automate due to the uncontrollable outdoor environment and variation. The agricultural practices are dependent on workers who handle tasks and operate machinery. The attempts to replace agricultural workers by robots have been slow [27]. Thus, automating farming tasks require dynamic and active interactions between the workers and the automation system to enhance workers and machines productivity. A collaborative interaction between human-robot and robot-robot guarantees a safer work environment and better quality [9].

2.2 Precision Agriculture

Primary production is the economy cornerstone for many countries. Consequently, primary industry has to maintain global competitiveness by employing smart, innovative and cost-effective technologies. Employing automation and robotics can be one of the best alternative solutions to current practices.

For example, primary products are NZ's main exports. NZ has approximately 14.39 million ha of farmable land. Two-thirds of the country land is made of grazing and grass landscapes. The total area of 10.63 million ha is used for sheep, beef, and dairy farming [10]. The success of NZ's economy which is mostly based on the success of primary production industries requires more productivity and efficiency. As a result, NZ has invested in innovative agricultural solutions and practices since the mid-1980s [28] to sustain their natural resources. The NZ

government focuses on remaining globally competitive but also being committed to maintaining a sustainable and green environment [10].

Farming technologies have become a profitable specialized industry. Technology-driven agricultural practices such as PA have enhanced productivity, profitability, efficiency, and suitability. PA's practices enable the right decisions at the right time to optimize resources and improve production. PA has a generic definition which has been evolving with technological changes [29]. It can be generally defined as agricultural practices which adopt technology to manage optimal and sustainable production. PA's practices have begun since the mid-1980s by employing available techniques to zone manage soil and crop variability. Later, new technologies were introduced such as microcomputers, the Global Positioning System (GPS) and Geographic Information System (GIS) which lead to Site-Specific Management (SSM) approach. SSM, which is a general term for PA, has adopted better technologies such as information technology and data analysis to manage farming, production, market, risk analysis, and finance [28].

PA decision making relies on gathering real-time information and measurements from fixed array sensors, satellites, planes, drones or ground vehicles [30]. Data collection allows farmers to target crops, farming animals and pasture specific needs with minimum resources and less environmental footprint [31]. The decision-making process is driven by a predictive approach, reactive approach or the combination of both. The predictive approach predicts crop specific needs and performance based on the yield history, spatial data records and variable indicators during the crop cycle, while the reactive approach relies on real-time monitoring and regularly updated data [32].

Several economies have adopted PA to increase eco-friendly and sustainable production while maintaining quality. New Zealand is one of the first countries to adopt PA's practices. In the

last forty years, PA's technologies have become one of NZ's specific industries which have contributed new technologies developed by local research facilities and universities. Since 2008, New Zealand has established eight Crown Research Institutes (CRIs) which were formed by joining several research institute and facilities to focus on researches' outcomes and avoid similarities [33]. The CRIs works closely with universities and research centres. One of these research centres is the New Zealand Centre for Precision Agriculture (NZCPA) at Massy University. The Precision Agriculture Association of New Zealand Inc. (PAANZ) was established in 2011 to connect users, companies, researchers, industries and students due to the growing interest and awareness of PA's benefits.

Given all of the above, automating agricultural tasks is a promising PA's practice which can provide specific and precise farming management. However, successful automation of some agricultural tasks is challenging and requires interactive systems. The implementation of collaborative robotics solves the automation complexity and enhances the efficiency of the resources. The collaboration of human intelligence with the machines information processing capability improves PA decision making.

2.3 Mechatronics and Robotic Systems in Agriculture

Farming has a laborious nature which is still dependent on workers. Workers have to work for long hours in farms, orchards, and greenhouses. The tasks performed by the workers are tiresome and repetitive. The workers sometimes work under harsh conditions, and they are subjected to hazardous materials such as pesticides and fertilizers [34]. Mechanization have been introduced since the early 1900s to power agricultural tasks, to overcome labour shortage, improve production and replace domestic animals used to power some tasks [35]. The use extensive of machinery in agriculture started during the World War II [36, 37]. Likewise,

research on autonomous vehicles started as early as the 1920s [24]. The first guided driverless tractors prototypes were introduced in the 1950s and 1960s [38, 39].

The decline of agricultural workforce nowadays is driven by many factors such as tasks' difficulties and non-ergonomic working conditions. Besides, most of the agricultural jobs are seasonal jobs which pay low wages. Thus, they cannot be relied on as a primary source of income [40]. Some agricultural jobs pay well, but people are not attracted any more to agriculture and farming jobs. On the other hand, other industries offer better employment benefits which attracted workers [41]. As a consequence, the current workforce is aging and the young generation are attracted to agricultural jobs provided that they have better education and better job opportunities [42]. For example, the number of farmers in Japan decreased from 4.82 million in 1990 to 2.60 million in 2010 [42].

Automation systems improve production quality, optimize resources and has less harmful effects on the environment. The information technology in automated systems process local and external data to execute complex operations [24, 25]. However, these technologies have been used moderately in agricultural sectors [21]. Nowadays, simple automation technologies are widely used in agriculture and food industry such as large scale plantations of cotton and grains; sorting, grading, and packaging; and meat processing. However, robotics and others advanced automation technologies are used merely in manufacturing.

For the last 20 years, researchers have managed to design and test several platforms and manipulator prototypes [27]. These developed systems were divide based on their functionality into 5 categories as follows:

1. Plantation robotics which performs harvesting, seedling, grifting, weeding, spraying, and transporting

2. Animal robotics which perform animal husbandries such as milking robots and wool shearing
3. Controlled agricultural environment robotics such as greenhouses robotics
4. Field robotics which manages quality grading, yield mapping, weed control, fertilization, and disease control
5. Post-harvest automation system which manage grading, sorting and packing. Currently, this technology is commercially available and fully automated [43].

The use of mobile machinery in PA was discussed by [25]. The paper highlighted the importance of performance and precision enhancement by employing high-precision sensors and actuators. Three systems were discussed which were grain yield sensors; yield mapping; site-specific spraying; and fertilizing. Their controllers and communication networks were also discussed. It was concluded that PA should have a standard communication system as a backbone to link weather services, traders, contractors, suppliers, farmers, and biological services. This paper has raised a valid point since agricultural industry is dynamic and cooperative information sharing improves decision making.

A personal to medium scale framing automation system was developed by [44] to perform seeding, watering, and weed removal. The designed Scalable Autonomous Agronomical Smartbot (SAASbot) was cost-effective and used rubber track to operate on non-flat services. The autonomous movement consists of a 3 axis Computer Numerical Controlled (CNC) mechanism which manages the farming activities and a mobile platform made of Mecanum wheels powered by DC motors to carry the CNC unit and move between farming cells. The system had a mobile application to interface with the user as an additional feature to the autonomous process. The application allowed users to remotely plant, water, and apply fertilizer. The system was tested and performed some tasks, however, the platform had a 3.1 mm error on the y-direction, 4.5 mm error in the x-direction, and 0.26 degrees orientation error.

Moreover, it experimented in a laboratory environment while the error would be higher if tested in an outdoor environment.

Mitsubishi Heavy Industry Company and Kyushu Electric built a fully automated vegetables plant factory [45]. A plant factory is an automated facility which produce vegetables and fruits with no or minimum human intervention [27]. Mitsubishi's factory provided a controlled growing environment and automated machines to perform a sequence of operations including seeding, seeds germinating, seedling nursery, transplanting, harvesting, and packaging. The factory produces 1500 vegetables per day. This factory have improved the production, but the company had to create whole new plant with a controllable environment similar to manufacturing plants. This is an expensive solution and only applicable for specific types of vegetables or fruits such as hydroponic crops.

Another system which was developed in Japan to fully automate large-scale plantations such as rice, wheat, and soybean [42]. The system employed three different design and functionality robots to automate rice production tasks including cultivation, plantation, and harvesting. The robots made of modified commercial machines were a tractor robot, transplanter robot, and harvester robot. The tractor robot which was a modified 47.8 kW YANMAR EG65 four-wheel-drive tractor performed tillage and puddled the paddy field. The transplanter robot which was a modified 7.7 kW KUBOTA SPU650 and 8.3 kW ISEKI PZ60 managed transplanting, fertilizing, and spraying. The third robot which was a modified ISEKI HF443, 31.6 kW was employed as a combine harvester. This study have proven that multi-agent homogenous systems are a feasible solution to collaboratively handle different and complex tasks.

The Agrob V14 is a mobile robot prototype developed to monitor the growth of mountain vineyard crops [46]. Mountain vineyards have a challenging environment and lack of accurate GPS signal or none at all. They also have challenging terrains which normally provide

imprecise odometer measurement. The platform prototype was made of a modified radio-control model Traxxas E-Maxx which was a 1/10 scale of a 4WD electrical Monster Truck. It was managed by a Robot Operating System framework (ROS) and its navigation was enhanced with a feature-based map built by identified natural and artificial features. A simulation environment was also created on Gazebo/ROS for testing and evaluation. The system successfully navigated using 2D EKF SLAM algorithm. This study highlighted the difficulties of navigation and localization in vineyards and provided useable solutions other than GPS.

2.3.1 Harvest and Yield Management Automation

Automating fruit picking with a robot or robotics manipulator can be successfully achieved in a sequence of three phases. These phases are identifying, locating, and reaching the fruit. The manipulator handles the last automation phase which is reaching the targeted fruit. Thus, identifying and precisely locating a fruit is critically challenging. A fruit by fruit harvest requires detecting each fruit; avoiding obstacles; and not to collide with the trees foliage, branches, or surrounding fruit.

The specialty crops such as fruit, vegetables, tree nuts, and nursery are manually managed. These crops still require traditional and skilled labour to handle intensive tasks. The current labor shortage and global competition is a real challenge facing these low scale production crops [26]. Full automation of specialty crops management is impractical due to surrounding variations. Their harvest management is labour-intensive and very difficult to automate since crops have ill-defined position, shape, colour, texture, size, and surface. There were several attempts to fully automate fruit harvest and the best 50 and most successful prototypes made in the past decade have never been commercialized [9]. These prototypes have an average localization success of 85%, average detachment success of 75%, average harvest success of 66%, average fruit damage of 5%, average peduncle damage of 45% and an average picking

cycle time of 33s [9]. Despite their poor performance, they were expensive to build or manufacture.

A sweet-pepper autonomous harvester was developed in the Netherlands [47]. The robot was made of two carts. The first cart carried a nine degree-of-freedom manipulator, controller, and a computer. The second cart carried vision sensors and an illumination unit. The system autonomously detected and harvested ripe peppers. The system was designed to pick sweet papers from V-cropped plants in a heated greenhouse. The robot managed to avoid damaging the peppers and their stem. It was tested in laboratory conditions, and 189 out of 194 fruit were successfully detected. The manipulator only reached 167 fruit and picked only 154 fruit. The overview cameras positive fruit detection rate was 0.87, and the false positive rate was 0.05 for 221 images and 479 images. However, the picking rate and cycle time were not discussed in the study or compared with other systems. The number of the picked fruit compared to the detected fruit was low due to the complexity of dynamic path planning when reaching the fruit.

The University of Almeria (Spain) developed a multi-use autonomous vehicle called Fitorobot [34]. The platform's dimensions were 0.7x1.7 m with a load capacity of 400 kg to safely and efficiently move between crop lines. It performed laborious and hazardous tasks such as spraying, platform lifting, and forklift transporting. A variable pressure based on velocity spraying system was utilized since it was not easy to maintain a constant speed due to the ground slope's variations and turning points. This system optimized the excessive use of chemicals. A petrol-combustion engine powered the platform and energized a hydraulic pump powering two hydraulic motors mounted to the rubber tracks. Two proportional control electro-valves controlled the steering mechanism. The platform had three control modes which were autonomous, remote, and manual control.

An autonomous Kiwi fruit harvester was developed by [48]. The kiwi tree has a vigorous vine which grows and spreads. The vine's growth is controlled and pruned regularly by humans to form a flat canopy supported by a structure. The flat canopy provides a shed to protect the kiwifruit from the weather and the sun. The kiwifruit hang down from above when growing, and can be easily picked without climbing ladders. An autonomous harvester was custom made to pick kiwifruit. It was made of an autonomous platform and four picking robotics arms. The platform autonomously navigated a preassigned path through the orchard while the robotics arms pick fruit and fill a carried bin. The platform deliver the bin when filled and fetch another empty bin. The platform was used also for pollination. The custom designed picking arms employed two types of asynchronous movements. This first movement was reaching the fruit and the second movement was enclosing the fruit envelop and rotating to detach the fruit with its stem. The arm took an average of 1s to reach and pick a fruit. However, the platform still have issues with navigation and has not been commercialized.

Apple detection algorithm using a Red, Green, Blue, and Depth (RGB-D) camera were investigated by [40]. The detection algorithm used both colour and depth information to detect, cluster, and separate apples from the foliate background. The developed algorithm used Euclidean clustering concept to the depth point cloud to identify the apple shape. The filtered shapes were then compared to a stored model by the Random Sample Consensus (RANSAC) algorithms. The obstacles were pre-defined as simple shapes and avoided. The results showed 100% successful detection of visible apples, 85% successful detection of partially visible apples, 1 cm localization error, and less took than 1s for detecting 20 apples. However, the system is limited to specific colours and illumination level. In addition, the detection rate would be less in real orchard in which fruit move and get covered by leaves during wind.

A designed and tested robotics prototype called Autonomous Fruit Picking Machine (AFPM) performed apple to apple picking and gripping mechanism [49]. The AFPM's gripper had a

flexible silicone funnel to suck selected apple; a camera placed inside the funnel using a vision system to locate fruit and a tractor tailing the platform. The system has a 6 DOF industrial robot (Panasonic VR006L); a generator to power up the platform; 2D horizontal stabilization unit; external vertical axis; touch panel Personal Computer (PC) as a central control unit; around canopy curtain to even light intensity; and a gripper. When the AFPM is in front of an apple tree, it scans the tree and divides into 40 look-out sectors. The apples were picked one by one from each sector. The fruit location was continuously updated while the gripper reaches them to enhance precision and to avoid damaging the fruit and the tree. The system detected and picked 80% of the apples with intact stem, and the remaining 30% had their stem pulled off. The average picking cycle time was 8-10s. However, a highly skilled picker takes 1–2s to pick a single fruit and pick nearly 99% of visible and non-visible fruit with their stem intact.

Automating apple harvest is difficult and requires smarter sensing technologies, better end effectors, and easier access or mobility within the orchard drive rows. Pickers have flexible movements and two hands that they can pick fruit delicately and fast. However, the pickers become tired, since they repetitively pick fruit and heavily walk to fill fixed bins. They also climb ladders to reach top fruit [23]. Automating fruit to fruit picking is inefficient, expensive, and unpractical. However, skilled labour is declining. As a consequence, enhancing worker performance is a practical solution. Labour enhancement is achieved by designing machinery or robotics platforms to aid the workers.

A mechanical harvest aid platform was designed to improve pickers' productivity by reducing most of the effort spent on laborious tasks [23]. The platform can be easily driven around small drive rows or trellises. The harvest aid platform allowed two pickers on the ground to fill their picking bags and empty the bag to a lower conveyor. Two more pickers could pick higher fruit using an elevated platform and place the fruit on the upper conveyor. A fifth worker was placed at the end of the receiving conveyor sorted defective fruit. The platform was driven to an

orchard row by joysticks and then could be switched to auto-steering mode enabling slow and safe autonomous movement. The harvest aid platform improved picking productivity by 22%. However, the system components caused an excessive fruit damage.

Apple vacuum-assisted harvest system was designed to overcome the difficulties of automating bulk or apple to apple harvest [50]. This system was designed to aid pickers and reduce fruit damage which might result from using a suction equipment. Two vacuum pumps provided suction were powered by an internal combustion engine. The fruit were sucked in when placed in return hoses leading to a proprietary deceleration mechanism to lower the internal pressure. The hoses were padded with a soft material to prevent fruit bruising or damaging. The system testing showed that fruit were not damages or busted and received a U.S. Extra Fancy grade.

Autonomous vehicles were designed to mow, spray, inspect disease or insects, estimate yield, and transport workers. The system aided harvest process by lifting up pickers and provided a safer working space compared to regular ladders [51]. Every vehicle was merely following the drive row to the end, exit, and enter another drive row to cover the whole orchard. The vehicles were made of a modified Toro eWorkman MDE electric utility vehicle and equipped with sensors, on-board computer, wireless modules, and a GPS module. The vehicle has three interfacing control modes which are manual, autonomous, and remote. The system increased top fruit picking efficiency by 58%. The workers who used the platforms indicated that the vehicles were safer and more comfortable than ladders. However, the vehicles had decentralized controllers and had zero cooperation with each other or with the pickers. They were supervised by the workers the entire time which made the using of automated system inefficient.

Bin management or yield management is another harvest task to be automated. Currently, full and empty bins are transported by tractors. A Bin-Dog prototype was designed by [52] to pick

up full bins and place empty bins. The platform navigated through drive rows and went over bins. The designed prototype had four wheels in which each wheel has an independent steering mechanism. The bin's managing cycle time was recorded 60 min since robot drove through the drive rows, loaded the bin, continued driving through and left the row.

2.3.2 Current Commercial Yield Management Solutions

Full automation of growing specialty fruit or even harvest is still far from being commercialized. Current commercial solutions are still dependent on workers. These solutions can only enhance workers' performance. Most of current commercial solutions are focusing on yield management.

The current yield logistics are handled by transportable fruit bins transported by forklift orchard tractors when filled by pickers. A standard size bin is 1200x1200x765mm. The orchard tractors can only transport one bin with a fork attached at the front. However, another fork was added to the back of the tractor to transport two or three bins at a time as shown in Fig. 2.1.

Fig. 2.1. An orchard tractor with front and back forks transporting full apple bins [40].

An orchard bin trailer is another standard method of transporting empty and full fruit bins. The trailer is towed to an orchard tractor as shown in Fig. 2.2. Multiple bins are transported by a single trailer and their capacity varies depending on the orchard size. The trailers are suitable to be used in large and flat terrain orchards, and they move along with the pickers.

Fig. 2.2. Three bin trailer towed by a tractor during kiwi fruit harvest [41].

The picking platforms are most practical and available solution, and they have been widely used in Europe. These platforms combine the fruit picking and transporting process. They are hydraulic self-leveling platforms which lift bins and pickers and provide safer working space. A signal platform can carry multiple bins depending on its specification to manage a continuous picking. It has self-loading bins mechanism powered by hydraulic rollers. As shown in Fig.2.3 pickers pick fruit at any height, fill their picking bags, and empty them to the bins on the platform. They provide a comfortable and safer workspace since pickers do not have to climb ladders and walk with full picking bags to bins. They limit the pickers' movement and flexibility.

Fig. 2.3. Hercules picking platform capable of transporting five full bins [42].

It is concluded from the studies mentioned above that fully automating fruit picking either by bulk or fruit to fruit is slow and unpractical. As a result, new solutions which focused on labour enhancement by using harvest aid machinery or robotics platforms achieved satisfactory results. These solutions rely on human physical flexibility and intelligence to overcome crops variations and complexity.

2.4 Cooperative Robotic Systems

Using multi-agent systems to automate complicated and more extensive scale agricultural tasks is a feasible solution. Modern agriculture management practices which are implemented to increase productivity are still dependent on labour which contributes to 60% of the total cost [13]. These methods are becoming more and more complicated. However, they can be easily automated since they are executed systematically. In fact, some agricultural products which are massively produced have been already automated such as cotton, corn and wheat. Other sectors such as orchards and greenhouses require more accurate and robust systems due to their complex environments and challenging working conditions [53].

A multi-agent system is a swarm intelligence system which is made of a team of agents effectively interacting with each other to complete specific tasks [16]. In a cooperative multi-agent system each agent collaboratively communicates with at least one agent while managing tasks. Thus, multi-agent systems can resolve complicated tasks which are difficult or impossible for a single agent to accomplish. A single agent's failure can also be compensated for by another available agent. The multi-agent system can improve the overall accuracy [54].

The swarm intelligence technology and multi-agent systems have had significant attention recently. However, many aspects and research areas remain to be explored [55]. The need for more sophisticated systems in industries leads to the emergence of multi-agent control systems [56]. The advantages of applying robotic multi-agent systems in agriculture are to redistribute complicated tasks into smaller tasks and achieve optimal and robust performance [17]. The required level of cooperation is determined by the application, participating agents, control architecture, and cooperative behaviour.

2.4.1 Robot-Robot Interaction

The exchange of data between robotic systems is referred to as a robot-robot interaction. Unlike humans who communicate with each other by voices or gestures, the robotics systems have two communication methods which are via data messaging or via visual observation [57, 58]. The data messaging method passes data packets which include information such as machine ID, pose, time stamp, velocity, and tasks' status. A successful message exchange is dependent on the team size and communication range, and the message can be passed via radio communication such as Bluetooth, IEEE 802.11x. (Wi-Fi), infrared, or 3G GSM internet.

Robots communicate messages or data through an inter-robot communication network. The inter-robot communication network is made up of both implicit and explicit communication methods. The implicit communication allows robots to broadcast essential data to the whole

system while explicit communication allows robots to establish a point to point communication. The use of implicit or explicit is dependent on the system, level of cooperation, and task [54].

A team of six cooperative autonomous vehicles is an example of robot-robot cooperative system [59]. The system performed reconnaissance and surveillance of a moving object including detection, tracking, and classification without human intervention. The system used ROS as middleware and a standard communication interface. ROS was integrated with a markup language called Battle Management Language (BML). An aerial guided vehicle (UAVs) and unmanned ground vehicles (UGVs) reached a destination point in a synchronized formation while the second team surveyed the point. The communication structure of the team was standardized through ROS and 3G mobile radio communication as an IP-based ROS. The tracking was done by defining the shape and velocity of a classified object. The team successfully moved in a robust formation to the desired point and autonomously surveil the proximity.

2.4.2 Cooperative Systems' Control Architecture

A multi-agent control architecture which manages the system behaviour is classified into three major topologies. These topologies are as follows:

1. Centralized control
2. Decentralized control
3. Hybrid control which combines the centralized and decentralized controls.

Selecting a suitable architecture for a system depends on the application, flexibility, and level of cooperation. The most widely adopted control architecture for multi-agent systems is the decentralized or hybrid control since they are more flexible and robust when compared to a centralized control system [60].

2.4.2.1 Centralized Control System

A centralized control system has an advanced robot or computer with a good processor employed as a master controller to perform global planning and manage a fleet. The master controller collects data from other robots, process the data, and executes algorithms. It plans tasks and manages the whole fleet on a global integrated level based on prior knowledge of each agent [61].

The advantages of a centralized control architecture are achieving optimal plans by maintaining data flow and feedback from every agent, the ability to predict system behaviour and results, and having direct control of the system. Nevertheless, if the central controller fails, then the whole system will fail. Also, the system or its participating agents could face intermittent failures due to the absence of communication coverage or being out of range [62].

Most of the common multi-agent systems used in agricultural applications have centralized control architectures. The most common architectures are in the form of a master to slave topology. This topology allows a sophisticated agent or computer to function as a master and the other robots as slaves.

A master-slave robot system was developed for harvesting and transporting hay on grassland by [63]. The master-robot controlled and planned the slave-robot path to a location-powered by developing a basic motion algorithm called GOTO which consists of the slave-robot's behaviour management instructions to find, follow, or predetermine a path for a point to another. The slave robot was also controlled to follow the master robot with an offset distance powered by FOLLOW algorithm which makes the slave-robot mimic the master navigation and predetermine the backward and lateral offset as shown in Fig.2.4. Both algorithms managed path planning, obstacles avoidance, speed, and steering control. The simulation results of both algorithms showed that a minimum risk index was maintained at 0.46. The

FOLLOW algorithm had an overshoot value of 0.134 m and 0.184 m RMS error when applying the sliding mode controller and PD controller respectively.

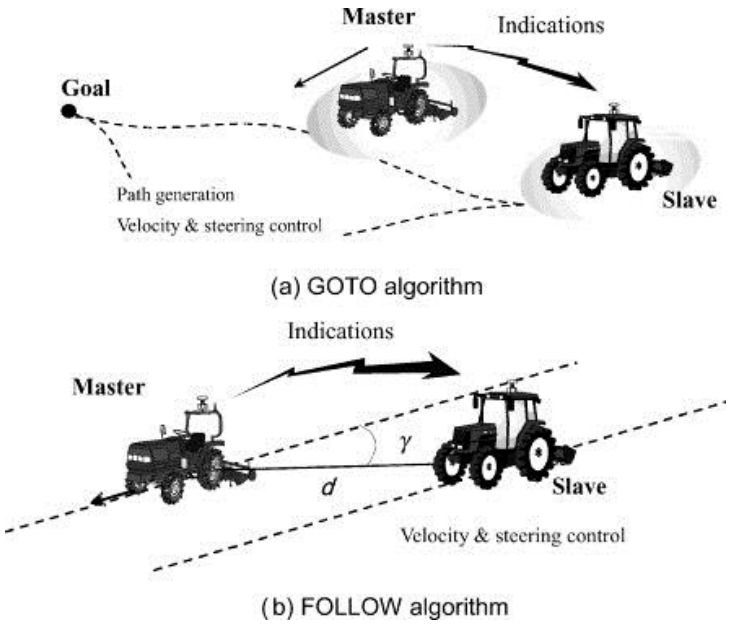


Fig. 2.4. Centralized Control System and the proposed algorithms for a master-slave robot system [54].

2.4.2.2 Decentralized Control

A multi-agent system with decentralized or distributed control system allows each agent to process its local data. Each agent makes its own decisions to achieve a common task. This control architecture enhances agents' autonomy, local data processing, communication coverage, tasks' planning, and independent decisions making [18].

Decentralized control architecture is comparatively robust against failures when contrasted with a centralized control system. Moreover, this type of systems breaks complicated tasks into smaller sub-tasks allowing smooth and fast response to dynamic conditions. The decentralized control systems have better cooperation behaviour [64]. However, the decentralized system is subjected to output oscillation which leads to inaccuracy and power wastage as a result of poor tracking. The system behaviour is unpredictable thus stability is not guaranteed [65].

A decentralized controlled smartweed treatment heterogeneous multi-agent system was designed by [53] as shown in Fig.2.5. The system is made of two unmanned aircraft system (UAS) and an unmanned ground vehicle (UGV) equipped with advanced vision sensors. A weed detection process was achieved by processing images obtained by a multispectral camera and range imaging time-of-flight camera. The data exchange mechanism allowed each agent to evaluate the overall task and handle their sub-tasks individually. The communication range limitation affected the data exchange process among the fleet. The UAS has a more comprehensive but distance observation while UGV has a closer but a narrower inspection. It was concluded that having heterogeneous multi-agent is more complicated but has better flexibility towards a wider range of applications and customized solutions.

Fig. 2.5. Heterogeneous and autonomous agents (from left): Vario XLC, Maxi Joker-3, and robuROC-4 by [44].

2.4.2.3 Hybrid Control System

A hybrid control system is a result of integrating centralized and decentralized control systems. The hybrid system has a two-level control hierarchical structure. The first level is a high-level control to master plan the system behaviour and monitor performance. The lower control level is the individual agents' controllers which forms a flexible and decentralized control. A hybrid system combines centralized and decentralized control systems' advantages and overcomes weaknesses [16].

A hybrid control system is suitable for applications which have complicated and data-intensive computations. These computations require larger memory to provide locality-aware

scheduling, fault tolerance, failure recovery, and load balancing. An advanced and powerful processor robot or computer collects data and keeps track of each agent. Each agent is decentralized control which could locally manage individual tasks such as navigation, detection, and mapping. The hybrid system can overcome pure centralized and decentralized structure limitations. Thus, it is a practical design for complicated multi-agent operations [66].

A hybrid cooperative control system for individual and fleet operating robots was presented by [67]. Autonomous vehicles were integrated with autonomous devices called implements carried or pulled by a vehicle to perform specific functions. The vehicles were a modified CNH Boomer-3050 which were equipped with a weed detection system, a navigation system, and a fuel cell to supply additional energy. The fleet is known as Robot Fleets for Highly Effective Agriculture and Forestry Management (RHEA). It was made of a central computer, human interface device, wireless communication, and a team of vehicles as shown in Fig.2.6. The system improved the reaction capabilities to speed changes and continuously improved trajectories. The tasks' processing time was optimized while four other processes were concurrently executed.

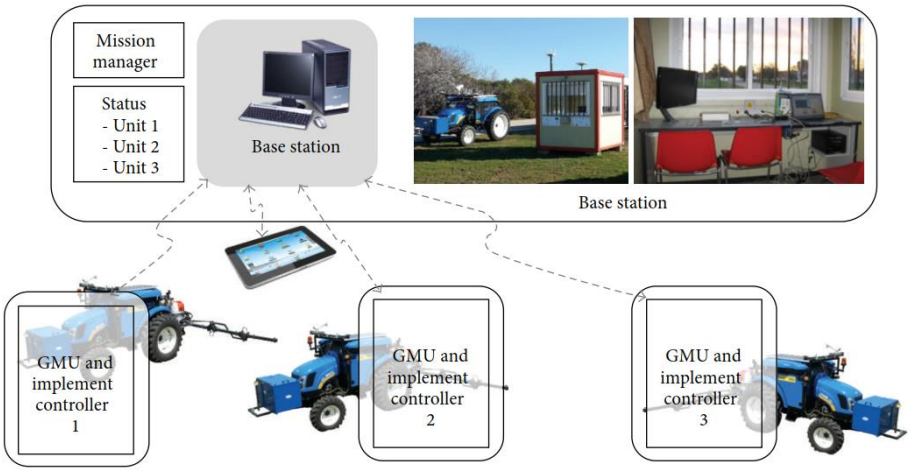


Fig. 2.6. Hybrid Control with a central base station and autonomous fleet vehicle with autonomous implements [46].

A heterogeneous multi-robot hybrid control system was designed by [68] which employed distributed algorithms for a team of mobile robots. The team shared spatial information among a highly equipped sensing robot with lower sensing ability robots. The system allowed localization and information sharing to obtain a combined estimation of agents' positions. The advanced robot helped less equipped robot and integrated all accessible information obtained from egocentric and inter-robot data, while each robot processes its local sensors data independently. An experiment was carried out by a team of three pioneer robots. All the robots A, B, and C performed odometer measurements while robot A and B performed features extraction from laser generated data. Robot C was effectively blind but had a gyroscope for a higher quality motion detector, but the ground truth and sub-centimetre accuracy position was obtained by recording and scan matching robots B and C positions. There was a lag on the information exchange since each robot was frequently generating the recent pose estimate. Robots C and B had unbounded error growth if travelled in isolation of robot A since it had a better map localization ability.

2.4.3 Cooperative Robotic Behaviour

Multi-agent system collaborative behaviours are determined based on their communication ability, data type, data intensity, agents' design similarities, tasks to be achieved, sensors' technology, and the control architecture. The degree of collaboration between robots specifies which type of exchanged information is needed. The simple collaborative system would only share spatial data, ID, and time stamp. On the other hand, complex systems with higher collaborative levels share data that contain commands, direction instructions, requests for a specific agent to do a specific task, or status updates [69]. The cooperation behavior was divided into three categories such as no cooperation, modest cooperation, and absolute cooperation [54].

2.4.3.1 No Cooperative Behaviour

A multi-agent system has a zero or no cooperative behaviour only if each participating agents communicate via implicit communication. The system is still collaborative since each agent share ID, position, and status. However, there is no cooperative behaviour since a point-to-point connection is missing. The homogenous multi-agent systems in agricultural applications such as spraying, planting, and fertilizer have zero cooperative behaviour [54].

A study on the safety of human-agents working with autonomous robotic agents was conducted by [70] to aid detection and tracking the present of human agents. The human-agent presence detection and tracking was done by calculating the distance between Ultra-Wideband (UWB) which reported the distance between two antennas on a robotic-agent and an antenna on the human-agent. The GPS was used for ground truth reference to evaluate the accuracy of the radio calculated and transmitted locations. A similar system was introduced by [71] to survey a perimeter with a team of mobile robots. A whole area is divided into none-overlapping sections. Each robot communicated its spatial data within a specific communication range. A decentralized control algorithm coordinated the robots based on the received data. The whole team collaborated to cover the whole area by monitoring each other positions and maintain their position and subtasks as shown in Fig.2.7.

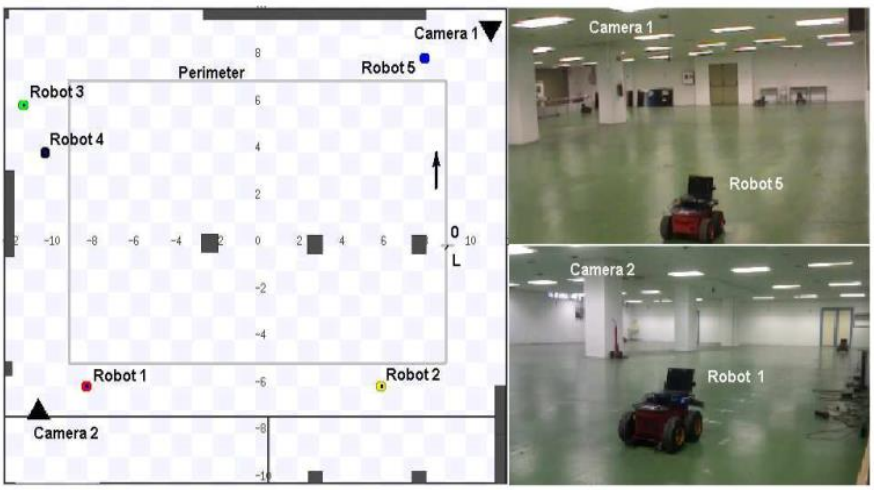


Fig. 2.7. A team of robots surveillance a perimeter collaboratively [63].

2.4.3.2 Modest Cooperative Behaviour

The second type of cooperation behaviour is a modest cooperative behaviour. The modest cooperative behaviour agents use implicit and explicit communication. The agents communicate via data messages. The implicit messages are broadcasted to every agent while the explicit messages are directed to a specific agent with specific data. The specific data are in the form of instructions such as requesting to dispatch a bale of hay, picking up fruit bins, or assisting another agent.

A modest cooperative system was designed by [72] which employed an autonomous vehicle as a transporter which was referred to as a supporting unit (SU) which supported another efficient autonomous vehicle referred to as a primary unit (PU). The system was simulated in MATLAB to perform fertilization in a paddock. The PU unit was simulated to perform spraying, and the sprayer tank needs to be refilled from time to time. The refilling process was done by the SU unit which arrives at the PU's location and refills its tank as shown in Fig.2.8. The PU unit sent a refill request message to the SU unit via an explicit communication.

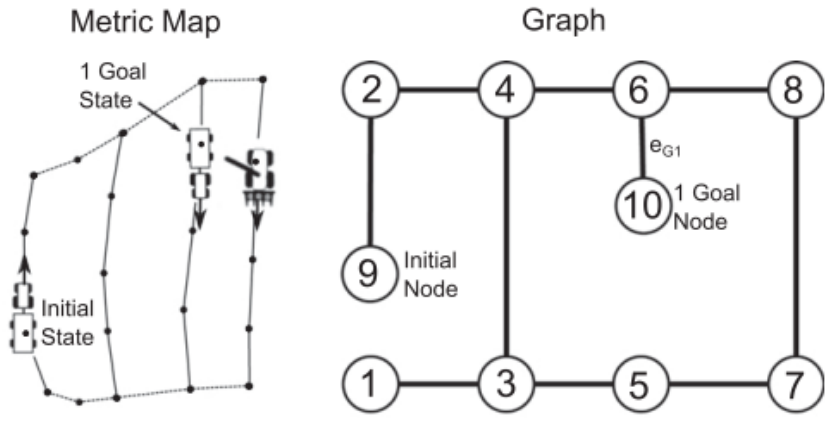


Fig. 2.8. The SU is arriving at the PU location to refill its tank based on PU request [64].

2.4.3.3 Absolute Cooperative Behaviour

A multi-agent system should have an absolute cooperative behaviour if tasks require continuous exchange of data. The participating agents establish a continuous point to point explicit communication besides implicit communication to exchange data.

For example, a multi-robot collaboration architecture was proposed by [73] to demonstrate an end to end grasping, autonomous transporting, and precise placement of interlocking components. Two agents were required to pick up beam components from storage, transport to a site, and place on a structure. The construction task was conducted on a sand pit to provide a similar out-door terrain in an unknown location and orientation but a known direction. The results of 25 runs to acquire components had one failure due to a poor wrist calibration. The results of 17 component placement runs had one failure due to lighting variability which caused noisy vision to find the fiducial location, but the results of 5 end to end runs had 0 failure. However, force-torque feedback was recommended to improve the positioning and to reduce vision sensitivity to adapt to outdoor light variance.

A multi-agent cooperative system to harvest grain was designed by [54]. The system has a hybrid control structure which employed a Central Processing Station (CMS) as shown in Fig.2.9. A Grain Harvesting Robot (GHR) accompanied by two autonomous grain wagon robots GWR I and GWR II had a decentralized controller. The GRH worked with one of the GWR I and they were always exchanging data. They were keeping a close distance to each other, so GHR could feed harvested grain to GWR I. Once GWR I's payload was full, a request was sent to GWR II to take over and receive the grain fed.

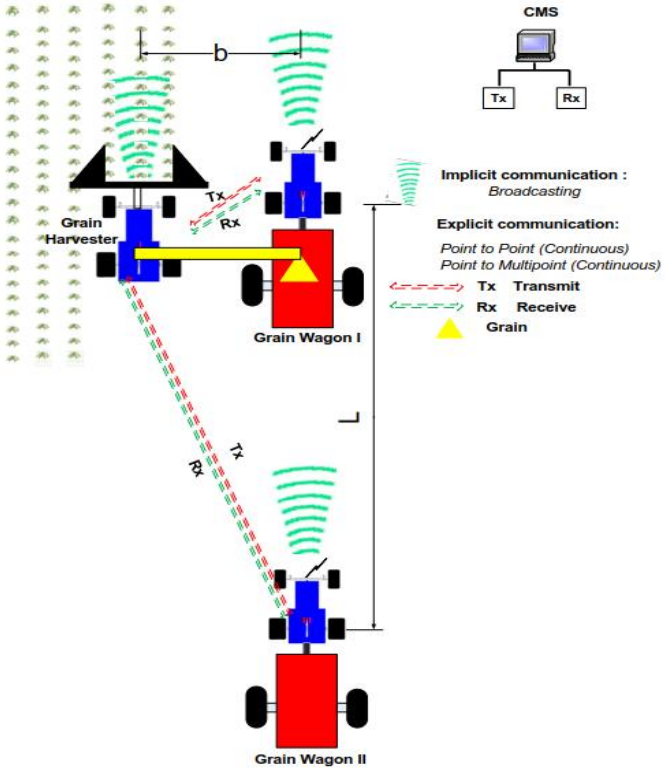


Fig. 2.9. A multi-agent grain harvest system with absolute cooperative behavior [44].

All things considered, collaborative multi-robot are more flexible and robust compared to a single robot. The cooperative behaviour between homogenous or heterogeneous robots allows a system to concurrently handle different tasks. The previously discussed studies highlighted the possibility of using a multi-robot system to handle complicated outdoor applications including agricultural tasks. The topology and level of cooperation were also discussed which concluded that a hybrid control multi-agent system with a modest cooperative behaviour is the most suitable architecture to automated harvest and yield management. Such a system collaboratively compensate failures to maintain a continuous process.

2.5 Human-Robot Collaborative System

Recent robotics technology enabled direct human-robot communication. The mean of human-robot communication can vary from being simple to complex depending on the application, robot technology, sensors, and the surrounding environment. Simple communication method

can be pushing buttons to make the robot executes simple tasks while complex communication includes voice commands or sign language instruction. Service robots are employed in crowded places, and they have been working with humans who are employed either as operators or participants [74]. It was necessary to make the robots adapt to the humans' presence and vice versa [74]. The human-robot communication creates a safer working environment around robots [57].

A position spatial (POS) ontology which is an extension of IEEE Robotics and Automation (ORA) Ontologies was experimented with the presence of a human agent by [75]. The experiment provided an understanding of spatial location data exchange between agents. The conducted experiment employed a transporter pioneer robot, an NAO H25 manipulator robot, and a human agent. The robots should pick up an object and find the human agent. Every agent's location is shared with the team to keep track of each other. It was concluded that the POS has to meet other requirements to be reliable such as a region to region data transformation and n-array spatial operators.

A heterogeneous context-aware ontology was designed to provide a personalized building tour by [76]. A collaboration between two robots employed as a receptionist and a companion improved a museum's guided tour. The ontology allowed a robot agent to mimic the human agent's behaviour by defining the human's interest to another robot guide. The two robots used for the experiment which were the Tank model as a receptionist and the CoBot model as a companion. The Multi-Robot Task Module (MRTM) was developed to resolve the multi-cooperative communication difficulties among heterogeneous robots. The interaction which was in the form of human-machine and machine-machine cooperation and optimal data flow improved the tour.

The latest advancement in information technology has improved robotics systems perception and decision making while interacting with humans. The perception of human presence and active human-robot interaction were initially developed to create a safe work environment. However, the current advancement in robotics technology has enabled the robots to collaboratively interact with humans to achieve common tasks. A collaborative multi-agent system benefits from human agents' intelligence and flexibility, at the same time it benefits form robots' precision, accuracy, and speed to achieve optimal results [18]. In regards to successfully automate harvest and yield management, it is very important to consider the human as active and cooperative participants to simplify the agricultural complex environment.

2.6 Summary

Agricultural tasks are difficult and repetitive; thus, they could be automated. On the other hand, the agricultural environment has complex dynamics, and it is automation unfriendly. Each agricultural activity requires designing a unique and customized system to automate them. Moreover, fruit picking is one of the most challenging applications for automation despite several attempts to design flexible and efficient harvesting systems. This review investigated the use of mechatronics and robotics systems in agricultural in which limitations were deliberated.

It was concluded that full automation of outdoors agricultural activities is impractical. However, the studies which focused on implementing automation to enhance labour have accomplished acceptable results. Moreover, the solutions which considered a collaborative automation approach have shown excellent flexibility and robustness while handling various and complex tasks. The possibility to incorporate collaborative automation systems in agricultural practices is feasible.

The most important factor to consider when automating agricultural activities is the human factor as an active and cooperative part of the automation system. The cooperation behaviour of a multi-agent system breaks down complicated tasks and improves accuracy. Current automation technologies have improved the robotics perception of human enabling a responsive and supportive human-robot and robot-robot interaction. Having evaluated all of the relevant studies, this study considers implementing a hybrid control topology multi-agent system with a modest cooperative behaviour to successfully automate harvest and yield management. A hybrid control architecture can manage a goal-oriented multi-agent system and improve the performed tasks overall accuracy.

CHAPTER 3: DESIGN OF A HYBRID CONTROL MULTI-AGENT SYSTEM

3.1 Introduction

Based on the finding of Chapter 2, this chapter presents the high-level system architecture required for a hybrid control multi-agent system to successfully automate the harvest and yield management. The system is designed to enhance the human pickers' efficiency by deploying a team of robotic transporting agents to handle the leg work and manage yield dispatching and transporting.

Agricultural production must be improved to meet future food demand and population growth. Automation is a necessary solution to optimise production and overcome labour shortage. However, automation is not an ideal solution since most of the agricultural practices cannot be automated easily. With current technology, the agricultural environment presents measurement challenges and variations which are unmanageable by current automation and sensing technology and the literature review highlighted that all the attempts to fully automate agriculture processes have not been commercialized except for sorting, packaging, and large-scale plantation crops.

Manual labour is still crucial for agriculture in general and specialty crops production in particular. The most practical and commercially available solutions are designed only to enhance labour performance. These solutions are mainly focused on harvest and yield management. The harvest and yield management are the most labour-intensive tasks. They require skilled labour in a short time period. In most of the developed countries, seasonal labour is outsourced and imported from neighbouring developing countries.

Currently, yield management solutions are employed to provide a safer, or more comfortable work condition. They combine more than one process to be handled by a single machine or a

platform. This machine can be used as an elevated platform to climb trees and as a carrier to transport bins. However, these solutions are still dependent on labour to carry tasks which are difficult to automate. This chapter aims to solve the current harvest and yield management automation problem by defining the limitations of current practices and designing a collaborative robotics solution. The system should enhance manual labour and optimize the cost of the overall system.

3.1.1 Problem Statement

An autonomous robotic solution that can collaboratively and safely operate in an orchard environment is required to automate harvest and yield management. The current horticulture harvest and yield management methods are well managed. Nevertheless, they are still dependent on human labour to pick fruit and walk to fill bins which are inspected by supervisors and transported by driven tractors to a sorting and packaging station.

This procedure is conventional, costly, and time-consuming. Moreover, repetitive walking to bins is exhausting and exposes workers to trip hazards. The cooperative nature of this system will overcome the orchard environment automation complexity and significantly enhance the harvest process.

3.1.2 Current Harvest and Yield Management Cost

Farmworkers such as tractor drivers and supervisors have an annual pay of 40000-53000 NZD. Moreover, they are also given extra benefits which include accommodation, food, transport, training, and bonuses worth 1000-4000 NZD [77]. Fruit pickers are paid 18-25 NZD per bin or paid 16.5-18 NZD/h [78]. The price for normal orchard tractor is 45000-80000 NZD depending on the model, size, and brand [79]. Operational costs of farm tractors vary based on their usage but it was estimated by [80] to be around 57.1 NZD/h (The cost was based on exchange rate of 1 USD to 1.45 NZD on the 3rd of Feb, 2019) excluding the labour cost since the drivers were

assumed to be working full time. The cost of treating the soil compaction, caused by tractors, is included in the tractors operating cost. Mounting a single forklift or an implement to a tractor will cost 41.62 NZD /h [80].

The failure rate of tractors depends on their usage and storage condition. For tractors stored indoors the average failure rate is 5.1-15.1/10^3 h while tractors stored outdoors have a failure rate of 13.8-17.7/10^3 h annually. These results were based on a study conducted by [81] who used a regression model to predict failure for a John Deere 3140 tractor. Therefore, John Deere 3140 Mean Time Before Failure (MTBF) was 67±7.92 hrs if stored outdoors and 155±6.33 h if stored indoors.

In conclusion, hiring a tractor driver and a supervisor; buying a single tractor; and paying 4 pickers wages only for 7 working days would cost 127924-198032 NZD considering the farmworkers annual pay, the tractor's price, and the wages paid for pickers while neglecting the implements and the tractors operating cost.

3.1.3 Solution's Performance Criteria and Constraints

The proposed automation system will minimize the harvest cost from the range of 130696-198032 NZD to 45824-84032 NZD or by 63% if considering the minimum cost estimate or by 58% if considering the maximum estimate. This cost reduction is achieved mainly by cutting out the tractor's driver and supervisor roles. Further cost, time, and soil compaction optimization are discussed in the following chapters. The automation system consists of a central controller, a fleet of autonomous mobile bin robots, and smart picking bags for the pickers. The central controller is a standard personal computer to master plan the harvest while the Smartbag monitors the picker. The system service waiting time should be 0-3s. The overall system reliability is the product of the number of employed platforms and a single platform's MTBF. The autonomous mobile bin robot specifications are as follows:

1. The platform will carry a fruit field bin to dispatch yield from pickers and transport to a collection station. It will replace the tractor and the bins.
2. The platform has the size of 800x800x300mm and the bin size is 700x700x600mm.
3. The bin capacity is 200 kg.
4. The platform speed is 2-4 m/s.
5. It is fully autonomous, safe, and simple to operate based on ANSI/ITSDF B56.5-2012 Safety Standard for Driverless, Automatic Guided Vehicles and Automated Functions of Manned Industrial Vehicles.
6. The platforms have an MTBF of 155±6.33 h similar to the indoor stored tractor.
7. The platform price is 60000-80000 NZD which is similar to the price of a new orchard tractor.

3.2 A Multi-Agent System to Automate Harvest and Yield Management

Automating the harvest process is difficult due to several technical and dynamic challenges such as entangled tasks, complex dynamics, crops variations, and challenging terrains. The review concluded that automating fruit by fruit or bulk picking is not practical and inefficient. Fruit picking is intricate and challenging to be automated due to the unique state each piece of fruit occupies on a single tree. Another challenging task to automate is post picking logistics referred to as yield management. The manual labour in agriculture is still essential. Thus, solutions which enhanced labour are more efficient and successful than fully automated systems.

It is essential to consider human factors when automating harvest. Hence, a cooperative behaviour between operating robots and workers is required. It is practical to employ a multi-agent system to manage fruit picking and yield management since it will handle the orchard's environment complexity. The system benefits from human picking agents' flexibility to handle

fruit picking and robotic transporting agents' self-localization, information processing, and navigation ability. Hence, robotic transporting agents are deployed as autonomous mobile bins that replace bins filled by pickers and tractors which transport the bins. This system can reduce pickers' unproductive walking time to fixed bins, thus improving productivity and performance. It can also automate the supervisor's duties which are monitoring workers, bins dispatching management, and payrolls.

3.2.1 The System Work Concept

A hybrid control collaborative multi-agent system is proposed to automate yield management. The system is designed to deploy autonomous robotics Transporting agents (RTAs) to serve human picking agents (HPAs) who are picking fruit. The system has a computer as a central controller. Every RTA has a decentralized controller the central controller is the master planner as shown in Fig.3.1.

Once the system has started up, a modest cooperative behaviour is established between RTAs, HPAs, and the central controller. The central controller monitors the HPAs' movement and fruit picking. The pickers' bag continuously broadcast location and weigh fruit. The RTAs continuously broadcast their location and payload weight as well as listening to information broadcast by HPAs and from the central controller.

The HPAs start picking and fill their picking bag. The smart bag sends a request for an RTA to dispatch the yield. The request includes picker's ID, position, and fruit weight. The central control assigns an RTA to arrive at the picker's position, and the picker empties their bag to the RTA's bin. Each RTA can serve several pickers before heading back to the drop station and deliver its full bin. The system keeps assigning RTAs to dispatch yield from HPAs as long as the harvest process is ongoing.

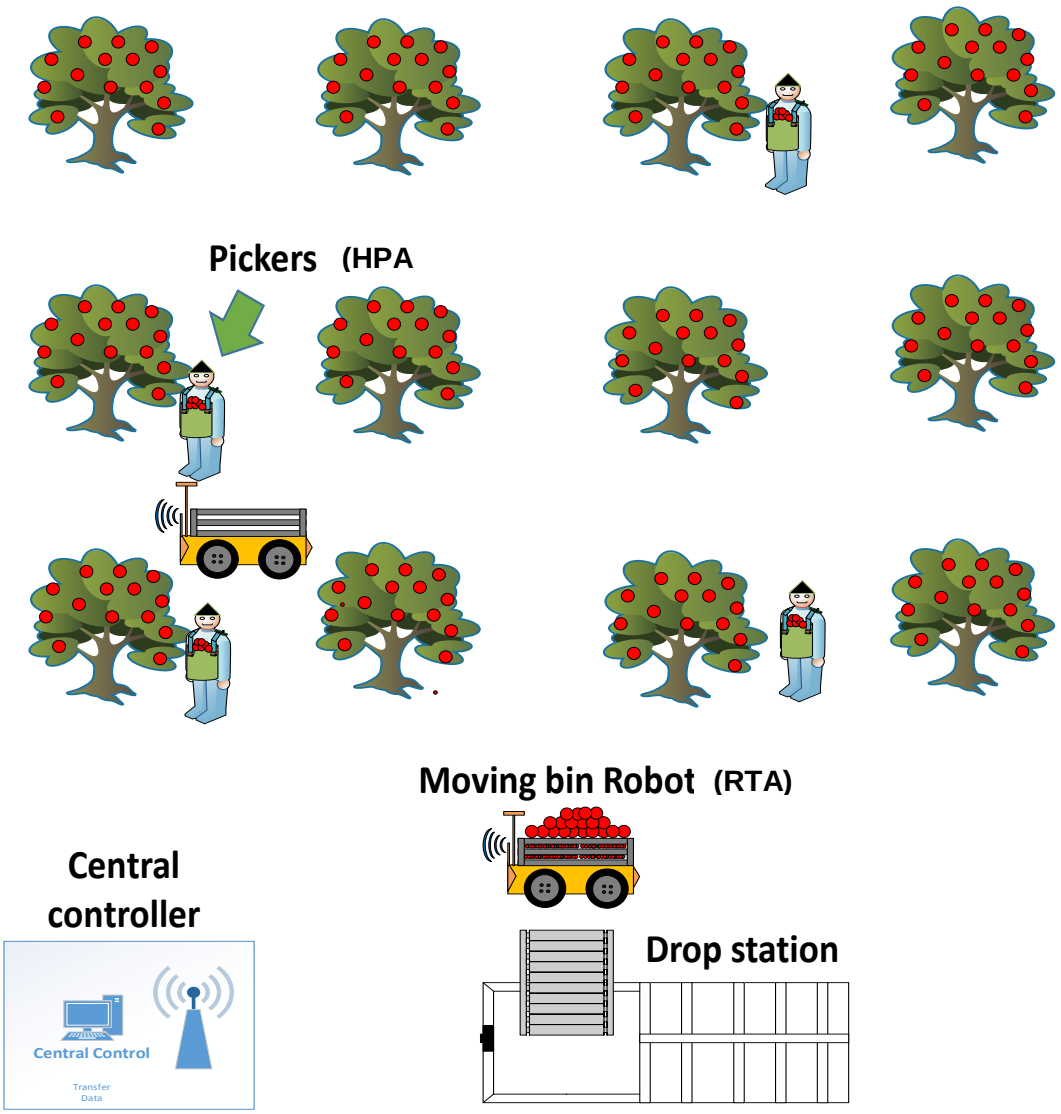


Fig. 3.1. A Hybrid Control Multi-Agent System to Automate Yield Management during fruit Harvest.

3.2.2 Multi-Agent System Architecture

The proposed system has a hybrid control architecture to manage collaborative behaviour between HPAs and RTAs. The hybrid control topology integrates the advantages of centralised and decentralised controllers. It is a two-level control architecture with a centralized control layer (Central controller) and a decentralized control layer (RTA's controller) is used as shown in Fig.3.2. HPAs' location and bag's weight data are continuously broadcasted by tracking and weight monitoring system implemented in their picking bags. The central controller manages transportation tasks by providing RTAs with instructions and optimal path. The RTA's

processor manages autonomous movement, navigates orchard's rows, avoids obstacles, updates trajectories, shares spatial data, serves HPA, and execute decisions independently to compensate the central controller's absence. The system communication medium is a Wi-Fi.

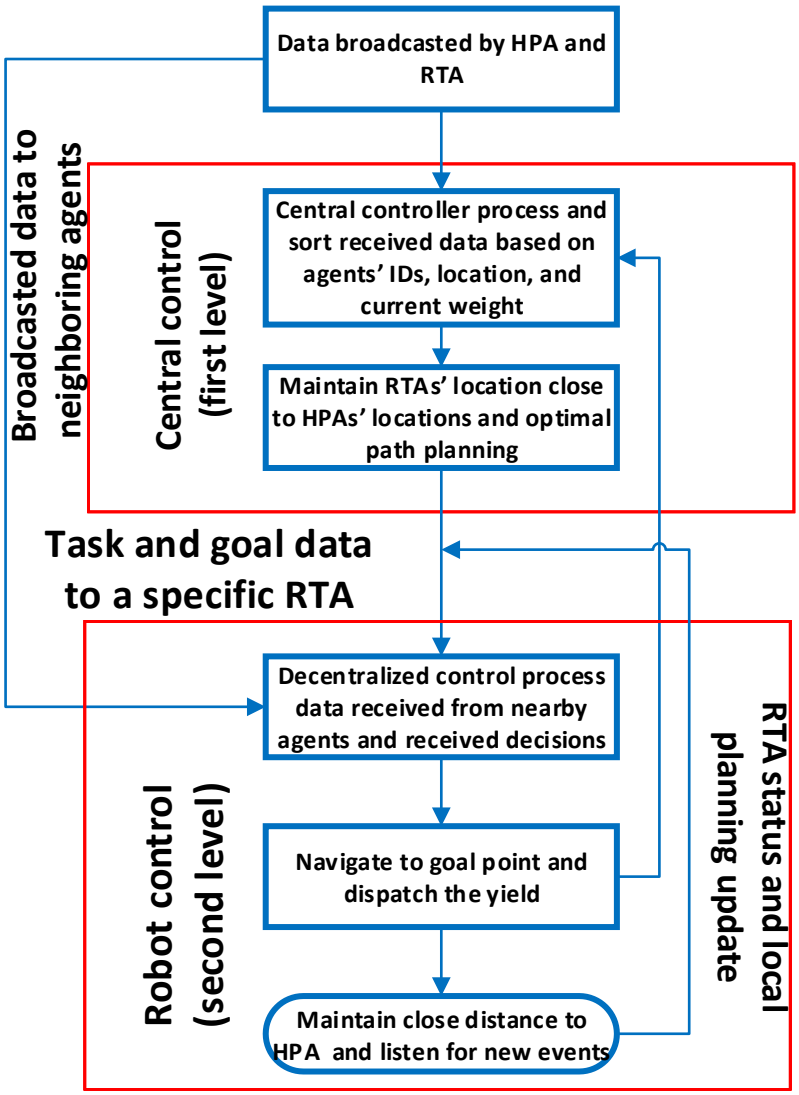


Fig. 3.2. Two layers control architecture, the top layer is a central controller, and the lower level is an RTA's controller.

3.2.3 Human Picking Agent's Tracking and Monitoring System Architecture

A GPS tracking device is implemented in the picking bag, and the yield's weight is also measured by two load cells. A microcontroller processes data and broadcasts data messages through a wireless module. The bag establishes a connection with the central controller and neighbouring RTAs. Once connected, it broadcasts HPA's ID and location. When the yield

weight reaches a specific threshold value, a dispatch request is added to the transmitted data as illustrated in Fig.3.3. The central controller and RTAs receive and process the request. The smart bag controller makes beeping sounds when the weight threshold value is reached and make another beeping sound to notify the picker of the approaching RTA. The smart bag allows picker to set and reset the weight threshold value as preferred within a recommended range. The pickers can place a manual dispatching request. Further details of the smart bag system can be founded in Chapter 7 which also discussed the system hardware design and field testing.

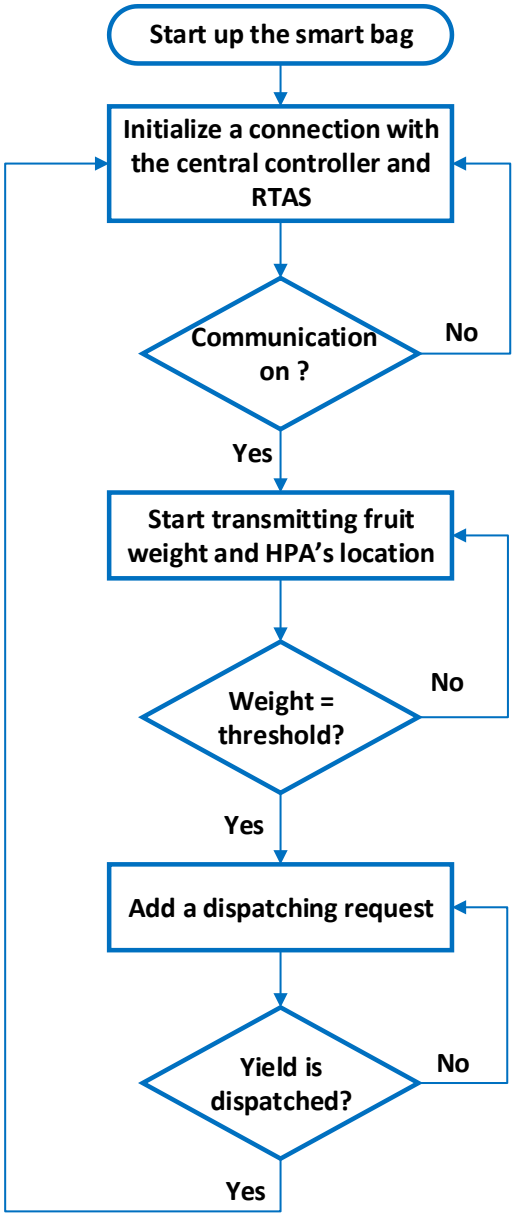


Fig. 3.3. Yield weight monitoring and HPA's location tracking system flow chart.

3.2.4 Robotic Transporting Agent Architecture

Each deployed RTA in the system establishes a wireless connection with the central controller and neighbouring RTAs. The RTAs receive and process data to update their local maps and keep track of other agents. The central controller processes dispatch requests and assigns a suitable RTA. The assigned RTA navigates to the HPA's location and dispatched the yield as shown in Fig.3.4. The RTA's controller processes its sensors data, navigates through rows, and executes local decision algorithms. After serval pickups, the RTA's bin is filled, and it is delivered to the drop station. After each pickup or delivery task, the RTA maintains a standby position and listen to coming events. The RTA responds to pickup's requests and transport bins independently during the absence of the central controller due to its decentralized control behaviour.

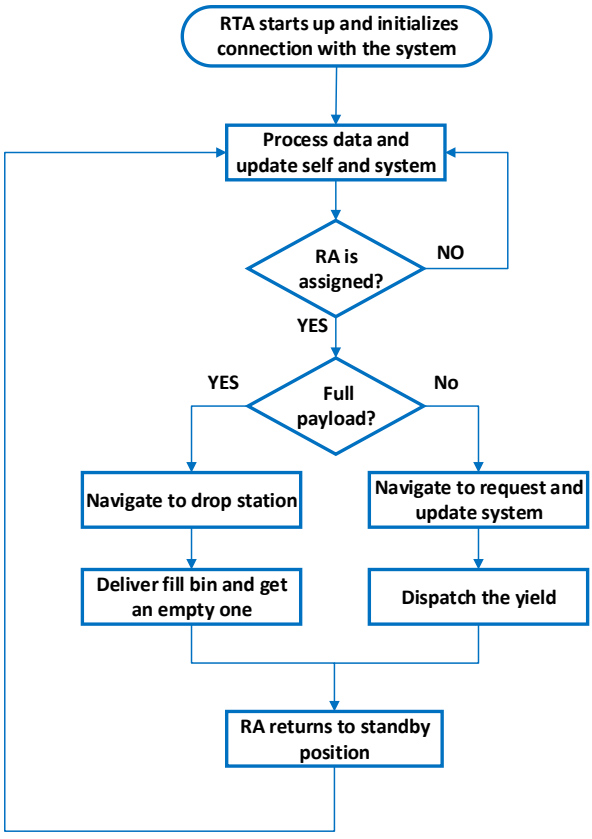


Fig. 3.4. RA as a decentralized controller flow chart.

3.2.5 Central Controller Architecture

The central controller is the master planner which receives, processes data broadcasted by every agent, and updates a global map. The global map is the orchard, and the central controller keeps track of each agent and updates their location on this map. The map is shared with RTAs to update their local maps and keep track of the current changes. The central controller assigns an RTA to dispatch HPA's yield when a request is made. Assigning the most suitable RTA based on its availability and location as shown in Fig.3.5. If all RTAs are busy or engaged, the central controller assigns the first available RTA by estimating their current task completion time. The central control finds the shortest path by employing the Dijkstra's algorithm [82]. The Dijkstra's algorithm is also used to estimate the arrival time and task's completion time.

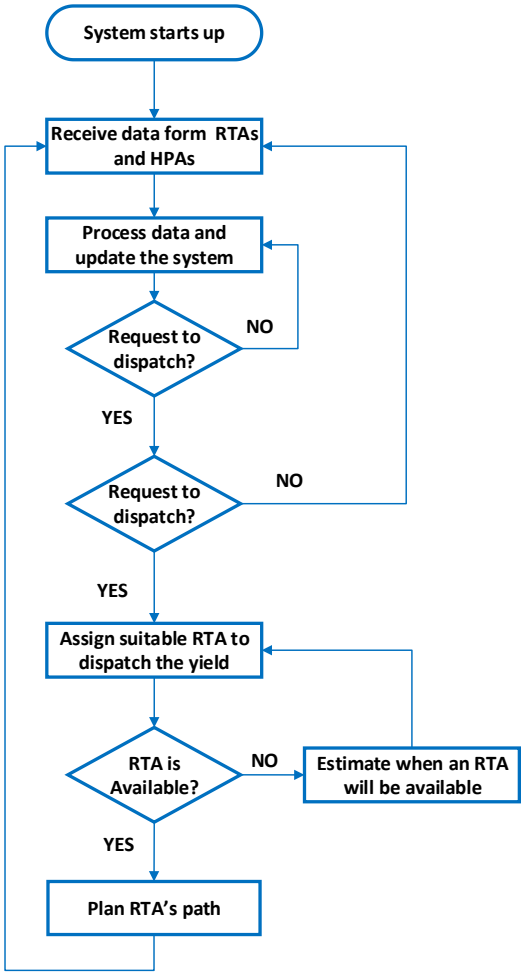


Fig. 3.5. Central Controller Architecture Flow Chart.

3.3 Summary

Automating fruit harvest and yield management is proven to be difficult and still depends on labour. A successful automation solution should consider labour enhancement. Previous studies concluded that automating the picking process is not practical and have a high cost. Thus, automating the yield management is practical and can enhance pickers' productivity. Automating the yield management requires a dynamic and active interaction between the workers and autonomous machines.

A hybrid control multi-agent system is proposed. This system has a team of agents which can effectively interact with each other to complete common tasks. This system has a hybrid control topology to manage human-robot cooperative interaction successfully. The system should be robust to failures since the failure of a single agent is compensated by another available agent. The overall system reliability is improved by increasing the number of operating RTAs. The RTAs can compensate the central controller failure as well.

The proposed system employs HPAs as pickers, RTAs as mobile bins, and a central controller which master plans the system behaviour. HPAs have a location tracking and yield monitoring device which establish an interactive communication channel between the picker and the system. This tracking and weight monitoring system is implemented within the picking bag.

The RTAs are deployed as mobiles bins which are assigned by the central controller to dispatch yield. They have decentralized control to enable safe navigation, local data processing, and decision making. They can plan and handle dispatching request in the absence of the central controller.

The central controller is employed as a master planner. It is the upper control layer and responsible for monitoring agents, assign suitable RTAs for yield dispatching, and ensures a

smooth and fast process. The pickers' data which are collected by the system are used for fair payrolls and provide further training and coaching enhancement.

The advantage of employing a collaborative multi-agent system to automate yield management is to redistribute complicated tasks into smaller tasks for optimal performance. The system has replaced the need for supervisor, tractor, and tractor's driver. It should improve picker productivity and safety since they do not need to walk to fixed bins. The designed system has considers worker sensitivity by ensuring a safe and less serving waiting time. The expected return of investment is high since the price of each RTA is similar to a tractor' price while the amount of 82000-114000 NZD can be annually saved by automating the tractor's driver and supervisor's roles. The RTAs are simple mobile robots since they do not have complex instruments or manipulators. The designed system satisfies the solution's performance criteria and constraints. Moreover, the system can be deployed to perform autonomous fertilization and precise spraying.

CHAPTER 4: MODELLING FRUIT HARVEST AND YIELD MANAGEMENT

4.1 Introduction

The proposed hybrid control collaborative multi-agent system for automating harvest and yield management, for which the architecture was presented in Chapter 3, has to be tested and verified. Thus, this chapter discusses the methodological modelling of fruit harvest and yield management methods and a novel method for automating the process. Modelling is a cheaper tool to test and evaluate the proposed multi-agent system than building a prototype provided that the concept is relatively new to agricultural applications. The objectives of the model are as follows:

1. Model a fruit orchard block with drive rows, tree rows, and a drop station
2. Model the behaviour of HPAs, tractors, and RTAs
3. Determine the fruit picking rate and bag yield rate
4. Determine the system service rate and service waiting time.

The proposed system has to manage the fruit collecting dynamics which present several challenges including HPAs' unpredictable behaviour. The orchard's environment variations and measurement uncertainties require intensive investigation. These investigations include collecting harvest data, conducting interviews, and running multiple experimental trails.

Fruit harvesting is a time critical and expensive process which must be conducted in a continuous but brief period. The harvest management includes hiring, training seasonal workers to pick fruit, and managing pre-harvest and post-harvest logistics including dispatching, transporting, storing, exporting, and marketing. Thus, orchardists carefully plan harvest to avoid profit loss. They do not like to disturbing the harvest which makes it is difficult to collect multiple sets of data repeatedly.

Moreover, the proposed collaborative system concept is new and has not been tested before in an agricultural environment. As a consequence, the uncertainties associated with this proposed method discourage the orchardists to risk loss of profit to conduct experiments during actual harvest given that apples are harvested once a year. Therefore, it is difficult, expensive, and time-consuming to conduct experiments and collect data.

A harvest and yield management model is required to understand the harvest process and test the proposed automation system. Thus, the model should enable a better understanding of the system's behaviour by simulating several experimental trails while and investigating possible methods to optimize the service waiting time and cost. The results are investigated to study the system response to changes. The results are visualized and analysed to explore other alternatives and enhance the optimization algorithms. Another advantage of the model is to view the system from different perspectives by running multiple simulation scenarios. The system's feasibility is determined by optimizing the harvest time and cost while enhancing the HPAs' performance and safety. To ensure a realistic model, the model should be based on current practice in a real orchard. The fruit harvest model in this chapter used apple fruit as an example to simulate the harvest process. However, the model can be used to model any fruit harvest by pre-setting the model's parameters to represent the selected fruit.

4.2 A Brief Overview of Real Life Apple Orchard Harvest Management Processes

A visit to Plant & Food Research Company's apple orchard in Motueka, New Zealand in June, 2016 investigated the apple growing and harvest process. The visit provided information about apple orchard layout, plantation, the harvest process, and yield management. The apple orchard was surveyed and mapped, and the orchard manager and workers were interviewed. The site visit report is presented in Appendix B1.

Harvest is the final and most crucial stage of crop management since it is the culmination of a whole farming year. Harvest management is consist of three stages which are pre-harvest, during harvest or yield management, and post-harvest as presented by the diagram shown in Fig.4.1. In the pre-harvest stage, workers are hired to pick fruit, supervise, and drive tractors. In the during harvest stage, pickers are organized in groups and each group is assigned to a specific block and has a supervisor. In the post-harvest, the fruit are graded, sorted, and sold.

This study focuses on the second stage which is during harvest. Workers manage several activities in sequence and parallel. These activities include fruit picking, transporting, sorting, and storing. HPAs pick apples and place them in small picking bags which they eventually empty into large fixed bins placed within the drive rows as shown in Fig. 4.2. Tractors transport full bins to a collection station for sorting and storage. These activates must be orchestrated systematically to ensure a smooth harvest at a minimum cost.

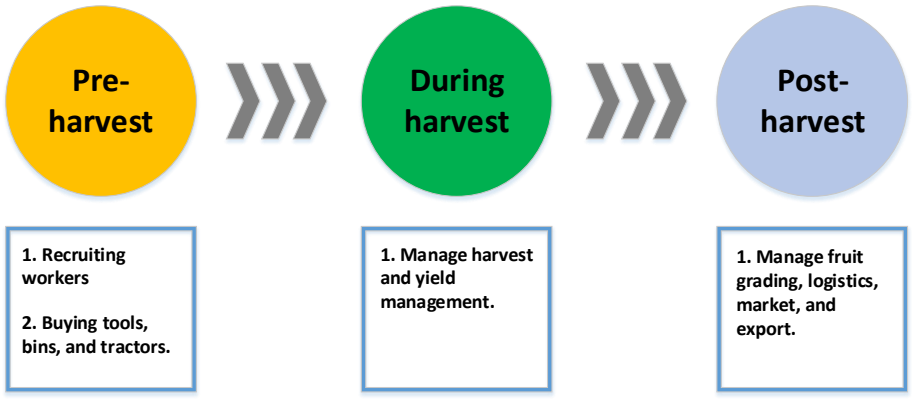


Fig. 4.1. Fruit harvest management diagram.

Fig. 4.2. Pickers are picking to a fixed bin to empty their picking bags [83].

4.2.1 Apple Orchard Layout

A standard size of a commercial apple orchard is 20-50ha. It is made of several blocks, and each block is 4ha. A block is surrounded by a pathway with a width of 5-7m which allows agrarian vehicles to access the block. Apple trees are planted in rows with an approximate spacing along the row of 2m while the drive row or space between the tree rows is typically 3m. The drive rows allow agrarian vehicles to move along the row as shown in Fig. 4.3. Apple trees are always pruned to maintain their height at 3m height for safer and easier picking. Wooden poles and stainless steel lacing wire also support the trees' branches.

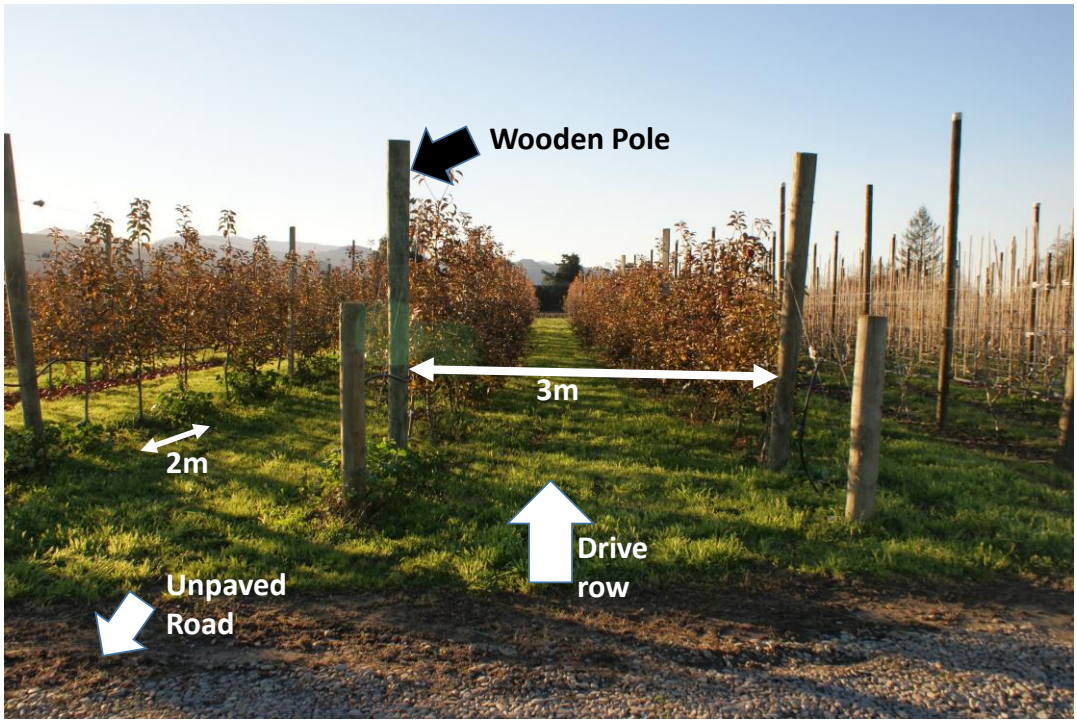


Fig. 4.3. Commercial apple orchard's rows showing trees, supporting poles, and steel wires.

4.2.2 Pickers Behaviour during Apple Harvest

Pickers traverse rows from end to end picking fruit methodically. Each picker starts picking apples from one side of a different tree in a row and moves from tree to tree until they cover all trees in a row. All rows in the block are traversed in the same manner. Pickers move in groups while each group is assigned to a different block. The size of a group varies based on the size of the block, however, a standard size block is 4ha while every hectare needs 2 pickers. Thus, there will be 8 pickers per group. Each group has a supervisor who monitors the pickers, estimates their wages per bin, and calls for tractors to dispatch and transport full bins.

Each tree's side is supposed to be visited once by a single picker. However, visiting a tree more than three times is impractical. Pickers pick apples to a picking bag strapped to their shoulders, use ladders to reach apples on the top of the trees, and walk to bins placed in the drive rows. A standard fruit bin has the dimensions of 1.22 x 1.22 x 0.8m and has the capacity of 300-350kg. The bins are placed within equal distances, however, sometimes they are moved along with the

pickers by a tractor. A picker produces 2-4tons of fruit per 8h work day depending on their experience and physical ability.

4.2.3 Orchard Tractor Movement to Transport Fruit Bins

Orchard tractors are used for several agricultural tasks such as transporting, spraying, cultivating, and towing. During harvest, orchard tractors dispatch and transport fruit bins. They deliver empty bins to the orchard rows before the harvest and transport full bins back to the drop station for sorting and storage.

A tractor lifts the bins from the ground by using a fork fixed to its front and back as shown in Fig. 2.1. Another bin transporting method is tailing a trailer which carries several bins as shown in Fig. 2.2. In this method, the tractor moves along with the pickers. Tractors are driven slowly to avoid damaging the soil, trees, and bins.

The visit to a commercial apple orchard provided a clear idea about the current harvest and yield management method. The information obtained either by observation, interviewing orchardists, and surveying provided understanding about the orchards environment and the harvest management. The data obtained from the visit enabled defining the parameters to build a harvest simulation model.

4.3 Modelling Harvest and Yield Management

Currently, harvest is managed by workers who pick fruit to picking bags, once they are full, they walk to a fixed bin to empty the bag and walk back to trees to continue picking. This method is inefficient, potentially hazardous, and prolongs the harvest. The proposed collaborative hybrid control multi-agent system aims to improve efficiency by deploying RTAs as mobile bins to serve HPAs. When the RTA arrives at an HPA’s location, the HPA empties fruit to the RTA’s bin. After several dispatches from different HPAs, the RTA delivers its full bin to the drop station. As a result, the HPA's traveling time is minimized.

Since the current harvest and yield management practices have been established during the site visit, an orchard environment can be modelled to simulate the HPAs and RTAs. The purpose of this model is to simulate fruit harvest by modeling a fruit orchard, fruit picking, tractor transporting bins and RTAs dispatching yield. The apple harvest was used as example to simulate the model. The simulation model will estimate fruit picking rate, picking bag yield rate, and RTA service rate. These results are used in a quantitative analysis to predict the system service time reliability. The service time reliability in this thesis refers to the possibility an RTA would service a dispatching request by a desired period of time. The quantitative analysis is using the Kaplan-Meier estimator and the queueing model to estimate the optimal required number of RTAs to achieve less than 1s waiting time in the system service queue.

Building the proposed system prototype for testing is expensive. Thus, building a simulation model is a cheaper alternative to run several experiments and analyse data. The results help to explain the process behaviour, improving design and optimization algorithms to control the outputs better.

4.3.1 Mathematical Model Formulation

Parameters and variables were derived by answering the following questions:

1. What are the available harvest data?
2. What are the system constraints?
3. What is the estimated time to fill a picking bag?
4. How many picking bags does a tree yield?
5. What is the picking bag's yield rate (the arrival rate of dispatching requests)?
6. How long does a tractor take to transport a bin to the drop station?
7. What is RTA's serving rate?
8. What is the optimal number of serving RTAs?

9. How long does a picker take to walk to a bin?

Fruit harvesting is modelled as discrete time events in which events are triggered and executed by the participating agents. The occurrences of events at particular times changes the system's state, and the system responds. Accordingly, the system's activities are simulated as parallel discrete events. For example, when an HPA starts picking fruits, a new event is started. This event ends when the HPA fills their picking bag and makes a dispatching request. This dispatching request triggers another event which is assigning an RTA to dispatch the yield. The dispatching request inter-arrival time and service waiting time are modelled as random variables. These random variables vary based on the HPAs' and RTAs' attributes. The model is simplified by considering the following assumption for an apple orchard:

1. Every apple tree has a random uniform distribution production of 350-400kg.
2. A fruit bin has a capacity of 350kg or can be filled by 17-25 picking bags depending on the weight an HPA could carry.
3. A picking bag has a capacity of 14-25kg which is pre-set by the HPAs before or during fruit picking.
4. Communication is ideal and uninterrupted during harvest.
5. RTA has constant speed which is not affected by the orchard's terrain.

The above assumptions were made base on the site visit to Plant & Food Research Company's apple orchard. During the site visit, the orchard manager was interviewed who provided information about the harvest process, production, and safety. The random uniform distribution was selected to model an apple tree production since it takes values which are equally distributed between the minimum and maximum production.

Each tree is assumed to produce a quantity of apples uniformly distributed between 350-400kg and b_w is the weight of a full bag or the weight threshold pre-set by an HPA. The random variable *B* is the number of fruit picking bags an HPA can fill from a single tree.

$$B \sim U\left[\frac{350}{b_w}, \frac{400}{b_w}\right] \tag{4.1}$$

Let e be the HPA's experience represented by a single apple picking time cycle in seconds while *w* is the total fruit weight an HPA could produce in 8h working day. The value a is a single apple weight which is uniformly distributed between 90-150g while n is the total number of apples picked by an HPA per day and t is the total work time in seconds of 8h working day.

$$n = \frac{w}{a}\ apples/day \tag{4.2}$$

$$e = \frac{t}{n} \tag{4.3}$$

Let E be a random variable which is generated from the uniformly distributed *e* values with the interval $[min(e), max(e)]$. This distribution is chosen since the fruit are located at different locations on a tree and each fruit takes a different picking cycle time which varies equally between the maximum and minimum speed values.

$$E \sim \mathrm{U}(min(e), max(e)) \tag{4.4}$$

Let $t_n$be the total time an HPA takes to fill a picking bag which is the sum of apple picking cycle time t_i for n apples/bag.

$$t_n = \sum_{i=1}^{n} t_i,\ provided\ that\ n\ changes\ based\ on\ weight \tag{4.5}$$

Let *d* be the Euclidean distance which HPA or RTA travels from point (x_1, y_1) to point (x_2, y_2).

$$d = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \tag{4.6}$$

4.3.2 Modelling an Apple Orchard's Block

In commercial orchards, apple trees are planted in rows and have a 2 m spacing along each row. Tree rows have a 3 m spacing to make a drive row for agricultural vehicles. Fig. 4.4 shows a 15 x 30m computer generated regular grid which represents a small orchard block. The coloured dots on the grid represent trees.

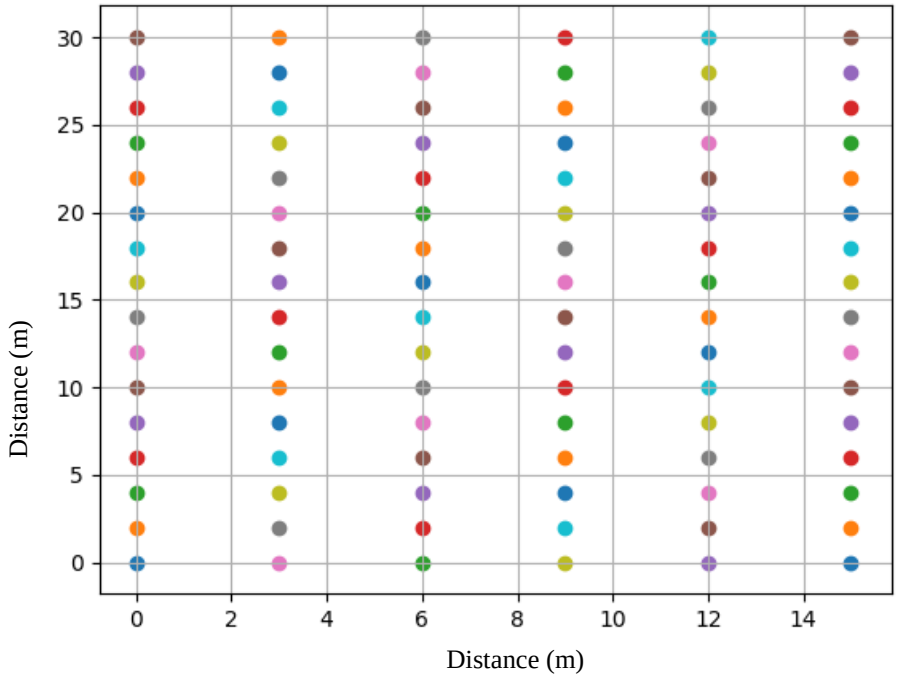


Fig. 4.4. A computer generated grid representing trees in an orchard block.

Apple trees are supported by poles and steel lacing wires which support the tree branches as shown in Fig. 4.3. As a consequence, movement between rows is restricted so pickers and vehicles can only traverse along the drive row.

A virtual block which represents a real orchard block was modelled to simulate apple harvest. The block was modelled as an undirected weighted graph to allow HPA and RTA models to move any point in the block following the shortest path. The undirected weighted graph let the

program execute graph algorithms such as the shortest path algorithm. Fig. 4.5 shows a simple undirected weighted graph representing the block shown in Fig. 4.4, which has 97 vertices connected by 106 edges. One vertex represents a drop station while the remaining 96 vertices represent apple trees.

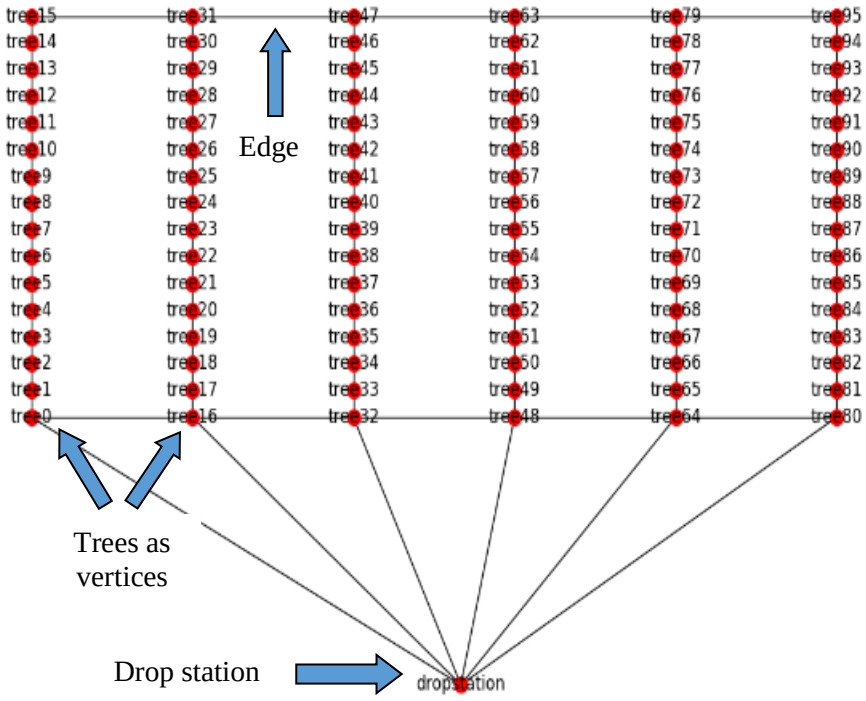


Fig. 4.5. Orchard block mapped as an undirected weighted graph.

For example, an RTA is located at tree63, and its bin is full after making several trips to serve HPAs, and it will head back to the drop station. Based on the graph above, there is a network of 6 possible paths the RTA could follow to reach the drop station. Dijkstra's algorithm [82] is executed to find the lowest cost path. In this case, it determines that the route starting at tree62 to tree48 and all the way to the drop station is the shortest path.

Dijkstra's algorithm creates a tree of short paths from the source node to the destination node. By weighing these in a graph, chooses the lowest cost path. The algorithm starts by initiating an array called 'queue' to hold all the graph's nodes, and initiates another array called 'distance'

which updates calculated distances from the source node to the following nodes on every iteration. At the same time, it initiates another empty array to hold nodes which were visited and de-queued from the queue array. In the end, the queue array will become empty, the visited nodes array will contain all the graph's nodes and the shortest path will be determined [84].

In the previous RTA's navigation example, the source node is tree63, and the destination node is the drop station. Dijkstra's algorithm adds all the nodes to queue array, removes tree63 form the array since it is the source node, adds it to the visited array initially, and sets all distances to other adjacent or neighbouring nodes to infinity. In other words, the algorithm marks all neighbouring nodes as unvisited. The source node *tree63* is connected to *tree62*, *tree47*, and *tree79* and during every iteration, each node is visited and marked with the sum of the distance to the source node. After visiting all the nodes connected to the source node, the algorithm starts exploring adjacencies in a similar manner. If a neighbouring node has a single adjacent node, the algorithm will proceed to its predecessor. On every iteration, the cost of each path is recalculated and updated until all the possible paths' costs to the designation node are calculated. The algorithm will then return the lowest path cost.

"NetworkX" is a Python package [85] which was used here to model orchard blocks as undirected weighted graphs and execute graph algorithms. A simple algorithm was developed to create regular grids which represent trees and drop station as shown in Fig. 4.2. The algorithm takes the block dimensions, tree spacing, and row spacing as inputs to create a grid which the algorithm converts to an undirected graph and connects its vertices with distance-weighted edges. The vertices are linked to each other vertically to form rows while the vertices at the beginning and end of each row are also linked horizontally to link rows as shown in Fig. 4.5.

The block modelling algorithm's flowchart is shown in Appendix A1, and the algorithm pseudocode is shown in Fig. 4.6. The block model takes the inputs and initiates a coordinates' array to store trees' coordinates which are generated by a nested loop. The nested loop has an outer *WHILE* loop and inner *WHILE* loop where the outer loop generates rows and triggers the inner loop which creates tree coordinates within the rows and appends them to the coordinates array. Another array is initiated to label trees which were created by a *FOR* loop. Afterward, an empty dictionary is initialized to store the trees' labels and coordinates. The algorithm executes a FOR loop to map trees' labels array elements as the dictionary's keys, and the coordinate's array elements as values. Finally, a regular grid is created which represents a block with labelled trees.

The algorithm converts the created grid to an undirected weighted graph by initializing an empty undirected graph and adding the dictionary element to become the graph's nodes. The nodes are linked by distance weighted edges in a way that represent the movement restriction through the trees' rows. The nodes are linked by a nested loop where the outer loop iterates the rows while the inner loop iterates the nodes on each row. There are three conditions within the inner loop which link each node to its neighbours depending on their location. The conditions are as follows:

1. If a node is at the begging of a row, it links with the node on the same row and the node on the next row.
2. If the node is at the end of the row, it links with to the last node in the next row.
3. If the node is within the row and not at the begging or the end of a row, it links with to the nodes on the same row.

Start orchard block modelling
Inputs: block_width, block_length, tree spacing, row spacing, and drop station's location
Outputs: Orchard block model

```
set (x,y) ← (0,0)
coordinates ← [ ]          // to create Cartesian coordinates for apple trees in the block//
while x ≤ block_width
        while y ≤ block_length
                append (x,y) to coordinates array
                increment y by tree spacing
        increment x by row spacing
        reset y to 0
trees_label ← [ ]                  // to label each tree on the graph//
 for i = 0 to coordinates' length
        append "tree" i to trees' label
 trees_coordinate ← { }        // will contain trees and drop station locations//
 trees_coordinate [trees_labels] = trees coordinates
 block_graph ← graph()          // initialize an empty graph//
 add trees-coordinate to block graph as nodes
 for a row in the block              // link the nodes together by weighted edge//
        for a node in the row
                if a node is at the start of the row & not at last row then
                        link to the neighboring nodes in same row and next row
                        calculate links' distances and add as weighted edge
                else node is at the end of row & not at last row then
                        link to the neighboring node in next row
                        calculate link's distance and add as weighted edge
                else node is not at the start or end of the row then
                        link to the neighboring node in the same row
                        calculate link's distance and add as weighted edge
 add a drop station to the graph as a node
 for a row in the block                  // link the drop station node the rest of the graph//
        for a node in the row
                if a node is at the start of the row
                        link the drop station to its neighboring node
                        calculate links' distances and add as weighted edge
End orchard block modelling
```

Fig. 4.6. Block modelling algorithm pseudocode.

Finally, the algorithm ends by adding the drop station to the graph and links it with the first node of each row. The drop station is linked to each row by a nested loop. The outer loop iterates through rows while the inner loop iterates through nodes on each row and only connect the first node on each row.

4.3.3 Modeling Human Picking Agents' Movement and Fruit Picking

HPAs pick fruit into picking bags which are strapped to their chest. When the picking bags are filled, an RTA dispatches the yield. A model of an HPA was developed to model fruit picking and the HPAs' movement during a harvest. For simplification, each tree is completely picked by a single HPA in a single visit. Each HPA picks fruit independently and has a variable picking rate, however, they share similar behaviour. Thus, the HPA model is represented as a class. The HPA models share the same modelling functions, but have different attributes. These attributes are the HPA's identification (ID), experience level, and maximum weight they can carry.

The HPA models traverse the rows and pick fruit in groups, with each HPA picking from a different tree in a row. When the tree is wholly picked, the HPA moves the next unvisited tree. The HPA models move systematically from a tree to tree along the row and then move to the next row until the whole block have been covered. Fig. 4.7 shows how an HPA moves while picking apples. HPA models start picking as a group following a picking path highlighted by the blue arrows in Fig. 4.7. Each HPA For example, if three HPA models were simulated, HPA1 would start picking at *tree0*, HPA2 would start at *tree1*, and HPA3 would start at *tree2*. If HPA1 finished *tree0* first, they would move to the next available tree which is *tree3*. The same goes for HPA2 and HPA3, whoever finishes first, will move to *tree4*.

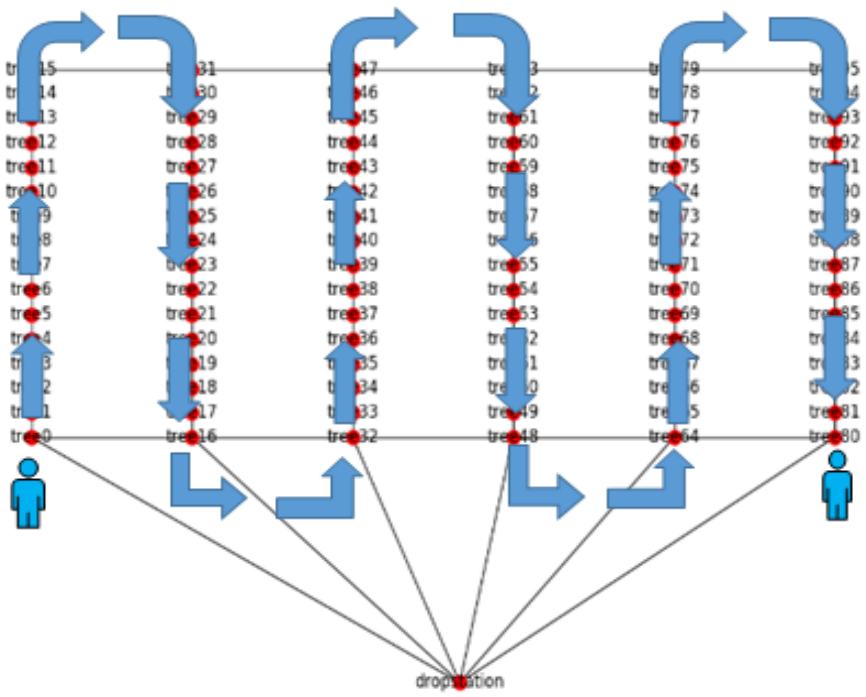


Fig. 4.7. HPAs' movement in orchard block during harvest.

The modelled HPAs algorithm executes a sequence of actions after initializing HPA models as class objects. At first, the algorithm imports a picking path predefined by a Depth First Search (DFS) algorithm [86]. The DFS algorithm is used as it returns a path similar to what a real picker would take as shown in Fig. 4.7. Every participating HPA's model starts picking fruit following the picking path. Once a picking bag is full, the model makes a dispatching request. The number of picking bags produced by each visited tree is calculated by Eq. (4.2). The apple picking cycle time is randomly generated by Eq. (4.5). The algorithm ends once all the trees in the block are picked.

The HPA modelling algorithm flowchart is shown in Appendix A2, and the pseudocode is shown in Fig. 4.8. The algorithm can launch multiple HPA models at the same time. The HPA models have different picking experiences which are a "beginner" HPA who takes 1.2-2.1s to pick an apple, "moderately skilled" HPA who takes 0.86-1.44s and "highly skilled" HPA who takes 0.65-1.1s. HPA models are executed concurrently while sharing the same environment which was generated by the orchard block model.

Start HPA's movement and fruit picking model
Inputs: HPA_name/id, HPA_experience, HPA_ bag_weight
Outputs: HPA model

```
1  import picking path array from block model    // determined by using DFS algorithm//
2   set max_no_bags_per_tree ← U(350 / HPA_bag_weight, 400 / HPA_bag_weight)
3   repeat
4          pop a tree from picking path array
5          move to tree location
6          set fruits_weight ← 0
7          for i = 0 to max_no_bags_per_tree
8                 while fruits_weight ≤ HPA_bag_weight
9                        increment fruits_weight by picked apple weight
10                request an RTA to dispatch the yield
11                reset fruits_weight ← 0
12         store tree's yield data to memory
13  until picking path array is empty
End modeling
```

Fig. 4.8. HPA's movement and fruit picking modelling algorithm.

HPA models traverse a picking path defined by a pre-ordered DFS. The DFS iterates the graph to visit and mark the nodes. In this model, it starts at *tree0* and traverses the graph by visiting all the nodes at least once. It creates two arrays which are a "marked nodes" array and "visited nodes" array to keep track of visited and unvisited nodes. After adding all the nodes to the marked nodes array, the DFS removes a node from this array on every iteration and appends it to the visited node array. Traversing the graph will stop when the marked nodes array is empty.

The picking path is imported to the model as a global array and each HPA model removes a tree from it. Each tree yields number of picking bags based on the uniform distribution defined in Eq. (4.1). The picking bags are produced by a *FOR* loop. The fruit picking is modelled as a *WHILE* loop which is nested within the *FOR* loop. The *WHILE* loop increments picked apple's weight to the bag's weight. The loop breaks once the bag's weight is more than or equal to the specified threshold weight value and makes a dispatching request. The trees are recursively popped from the picking path array. Once the picking path array becomes empty the fruit picking model ends. Obtained data are stored in the memory for analyses and algorithm improvement. The data represent the number bags produced by every tree, the weight of each

bag, the time is taken by HPA model to fill a bag, time to completely harvest a tree, and service waiting time.

4.3.4 Modeling Harvset and Yield Management Automation

RTAs replace fixed fruit bins which are walked to and filled by HPAs, inspected by supervisors, and transported by tractors. RTAs are assigned by the central controller to serve the HPAs. The RTA delivers its full bins to the drop station after serving multiple HPAs. Each RTA has a decentralized controller to control its behaviour compensate for the absence of the central controller. Each RTA is independent, but it continuously exchanges data with other agents and the central controller.

The RTA model algorithm models a mobile robot bin. The model responds to dispatching requests and then delivers its full bin back to a drop station. The request is made by the HPA when filling their picking bag. The arrival of a dispatching request is when full picking bag is filled by an HPA so the picking bags yield rate is the dispatching requests arrival rate.

The central controller also receives the dispatch requests. If the central controller is available, it will assign a suitable RTA to dispatch the yield. However, if the central controller is absent, the nearest available RTA will respond to the request. The assigned or responding RTA navigates to the requester's location by following the shortest path which is determined by Dijkstra's algorithm [82]. The central controller algorithm select and assign the RTAs based on one of the three dispatching algorithms which are:

1. Available First Serves (FAFS) dispatching algorithm
2. Dynamic Distance (DD) dispatching algorithm
3. Dynamic Distance and Best Fit (DDB) dispatching algorithm.

The dispatching algorithms are discussed in Chapter 6. The RTA model's algorithm flowchart is shown in Appendix A3, and the pseudocode is shown in Fig. 4.9. The algorithm models

multiple RTAs at the same time which share the same environment with HPA models. Each RTA model is initialized as an independent resource and has its attributes which are ID, location, speed, capacity, and payload. The algorithm sets an empty dictionary to map RTAs' IDs as keys and their locations, speeds, capacities, and payloads as values. The dictionary is created to track each RTA's location and payload. An empty array for RTA availability is created to store the RTAs' IDs. Another empty request array is created to enqueue dispatching requests. Once there are dispatching requests in the queue, the algorithm assigns an available RTA to dispatch yield from the request's location.

The model algorithm responds to the request by executing a nested loop. The inner WHILE loop checks RTA availability array, and the outer WHILE loop checks the available RTAs' payload. The dispatching algorithm chooses an available and suitable RTA which moves to the request's location, dispatches the yield, and increments the yield weight to the RTA's payload while updating the dictionary values. Then, the RTA's ID is appended back to the availability RTA array. If the RTA's payload is greater than or equal to the RTA's pre-set payload threshold, it will head back to the drop station. Once the bin is delivered, the algorithm resets RTA's payload to zero, moves RTA closer to HPA, appends RTA's ID to the availability RTA array, and updates the dictionary. The modelling algorithm ends once the harvest process is over and the HPAs make no more dispatching requests. The process data are stored in memory which include the bag's yield rate, and the RTAs serving rate.

Start modelling RTA as a mobile bin
Inputs: RTA_ID, RTA_capacity, RTA_speed, and RTA_payload
Outputs: RTA model

```
RTA_dictionary ← { }
RTA_dictionary [RTA_ID] = RTA_capacity, RTA_speed, RTA_location, and RTA_payload
available RTA ← [ ]
add RTA_IDs to RTA_availability_queue
request_queue ← [ ]
add dispatching requests to request queue
repeat
        while RTA_availability_queue is not empty
                dispatching algorithm chooses a suitable RTA
                while RTA_payload < RTA_ pre-set payload threshold
                        remove a request from the request queue
                        move to request's location and dispatch yield
                        increment yield weight to RTA_payload
                        update RTA's location and RTA's payload at RTA dictionary
                        append RTA_ID to RTA_availability_queue
                feedback as busy
        move to drop station
        deliver full bin
        reset RTA_payload to 0
        update RTA_location and RTA_payload at RTA dictionary
        store bag dispatching time or HPA's waiting time to memory
until no more requests or request array is empty
```
End modelling

Fig. 4.9. RTA as a mobile bin and yield dispatching modelling algorithm.

4.3.5 Modelling Current Harvest and Yield Management

In a real life scenario before fruit picking occurs, tractors transport empty bins to the orchard drive rows. These bins are transported back to the drop station after being filled by pickers and inspected by supervisors. There are several methods as to how tractors deliver bins as mentioned in Section 2.2.2. The most common method is lifting the bin with forklifts mounted at the front and the back of a tractor.

The proposed multi-agent collaborative hybrid control system automates harvest to cut out bins, supervisors, and tractors. However, the system's performance has to be validated

according to well defined objectives. Verification is achieved by comparing the simulated results against a current harvest and yield management method model.

For simplification, it was assumed that the tractor has a constant speed, moves on obstacle free paths, and transports a single bin at a time. HPA models walk to fixed bins every time they fill their picking bags. The orchard block model in Fig. 4.3 was modified by adding bins as graph's nodes as shown in Fig.4.10. Each bin is linked to two trees with distance weighted edges. The new block model has 96 trees and 48 bins with a total number of 145 nodes including a drop station. The bins are added to the block model by adding two algorithms to the original block modelling algorithm. The first algorithm is to add the bins locations as Cartesian coordinates to block grid and then convert them to graph nodes while the second algorithm links them to trees.

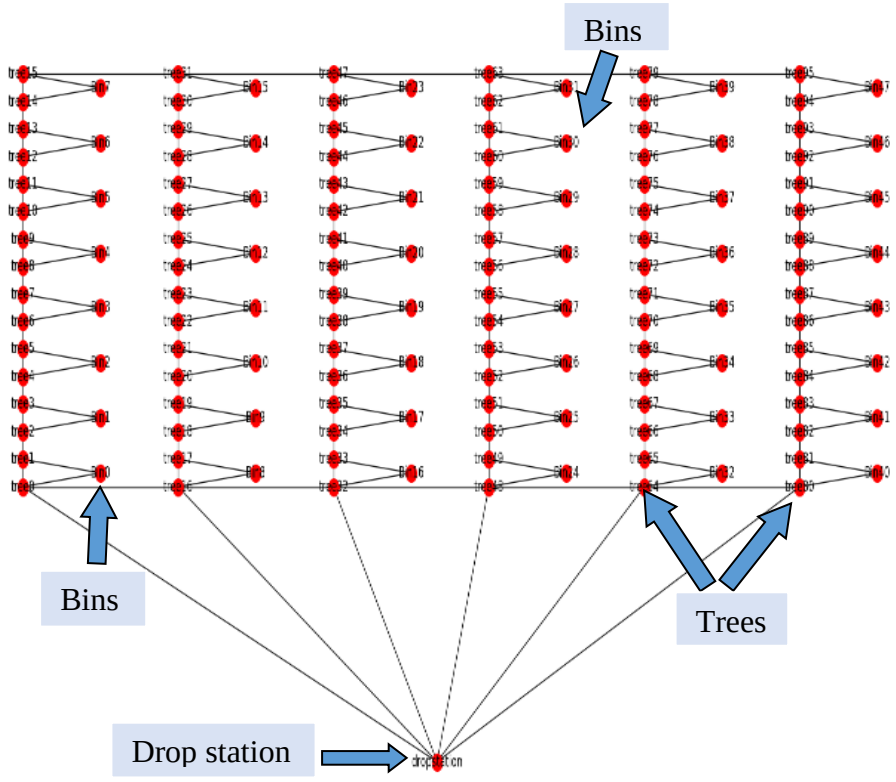


Fig. 4.10. Orchard block model with bins on the drive row.

The bin adding algorithm flowchart is presented in Appendix A4, and its pseudocode is shown in Fig. 4.11. The bins are added to the graph in the same way trees were created in the original block model. A spacing between bins is the algorithm input which creates an empty array to store bins' coordinates generated by a nested loop. The outer loop generates bins' rows while the inner loop creates bins' coordinates. Another array is created to store bins' labels. Afterward, an empty dictionary is initialized to store the bins' labels and coordinates. The algorithm executes a FOR loop to map the labels array elements as the dictionary keys and the coordinates array elements as values. The dictionary elements are added to the block graph as nodes. Finally, the bins nodes are linked to every two trees by a FOR loop. The FOR loop iterates the nodes indices in the graph and link each bin to the corresponding trees.

Start adding bins block modelling
Inputs: new_width to block_width plus 3
Outputs: orchard block model with bins

```
set (new_x, new_y) ← (1.5, 1)
bins_coordinates ← [ ]          // to create Cartesian coordinates for bins in the block//
while new x ≤ new_width
        while new y ≤ block_length
                append (new_x, new_y) to bins_coordinates
                increment y by tree_spacing plus 2
        increment new_x by row_spacing
        reset y to 0
bins_labels ← [ ]  // to label each bin on the graph
 for n = 0 to coordinates' length
        append "bin n" to bins' label
 bins ← { }
 bins [bins_labels] = bins_coordinates
 add bins to block graph as nodes // next is to link the nodes together by weighted edge
set bin index to the length of tree coordinates plus 1
for m = 0 to max_column index + 1
        set node a to graph node[2 times m] // assign a specific node in the graph by its index
        set node b to graph node[2 times m plus 1] // assign a specific node in the graph by its inde
        set bin to graph node[bin index plus m] // assign a specific node in the graph by its index
        link node a and node b to bin
        calculate links' distance and add as weighted edges
```

End adding modelling

Fig. 4.11. Adding bins to the block model.

The HPAs modelling algorithm was also modified to allow HPAs to walk to the bins. The modified algorithm now models HPAs walking to bins to empty their picking bag, then back to a tree to resume picking. The HPAs graph traversal still follows the same predefined picking path. The algorithm calls a tractor to transport full bins back to the drop station. The bin filling and tractor calling were simplified by assuming that every two trees fill a bin.

The modified HPA model algorithm flowchart is shown in Appendix A5, and the pseudocode is shown in Fig. 4.12. The algorithm allows HPA models to walk to bins. A WHILE loop models fruit picking while the algorithm moves the HPA to the nearest bin when filling a picking bag. When a bin is filled with fruit, the algorithm sends a tractor to transport the bin.

This process repeats until all the trees in the block are visited. The data are stored in memory for data analysis. The stored data are the number of produced picking bags per tree, bag's weight, time to fill a bag, time to completely harvest a tree, HPA's walking time to a bin, and bin's transportation time.

In both the tractor and the RTA models, the HPA modelling algorithm has a fixed picking bag threshold value which allows the HPA to walk to a nearby bin or request an RTA only if the measured weigh is equal to or more than the weight threshold value. For example, the HPA model would empty the bag only if their bag is full. In some cases which is highly unlikely to happen, the HPA model would move to the next tree and pick few more apples before making a dispatching request or walking to a bin.

Start HPA's movement to fixed bins and fruit picking
Inputs: HPA_ID, HPA_experience, and HPA_bag_weight
Outputs: HPA model

```
1    import picking path array from block model    // picking path array
2    set max_no_bags_per_tree ← u (350 over bag weight, 400 over bag weight)
3   repeat
4           pop a tree from picking path array
5           move to tree location
6           set fruits_weight ← 0
7           set tree no ← 1
8           for i = 0 to max no. bags per tree
9                   while fruits_weight ≤ HPA_bag_weight
10                          increment fruits_weight by picked apple weight
11                  walk to a bin to empty the picking bag
12                  reset fruits_weight ← 0
13          if tree no = 2
14                  call tractor to transport the bin
15                  rest tree no ← 1
16          increment tree no by 1
17          store tree's yield data to memory
18   until picking path array is empty
End modelling
```

Fig. 4.12. HPA's movement and fruit picking modelling algorithm while employing a tractor.

A tractor model pseudocode, which models bins transporting, is shown in Fig. 4.13 and the flowchart is appended at Appendix A6. The tractor model only moves when a bin is filled. The algorithm makes a call for a tractor to transport the bin. The tractor model picks up the bin and transports it back to the drop station. This process is repeated until the harvest is over. The data are stored in the memory which include the bins' labels and transportation time.

Start tractor as bins transporter model
Inputs: tractor_speed
Outputs: tractor model

```
1    request to transport a bin
2    repeat
3           move to the bin's location
4           pick up the bin
5           move back to the drop station
6           save data regarding bin's location and time to transport to memory
7   until all bins are transported back to drop station
End modelling
```

Fig. 4.13. Tractor modelling algorithm.

The modelled orchard block serves as a virtual environment to simulate the current harvest and yield management method and to simulate the proposed method. The current method can now be simulated in an orchard block model, which has bins within its rows, in which the HPA models pick fruit and walk to fill the bins while the tractor model dispatches and transports full bins. Moreover, the proposed automated method can also be simulated in an orchard block model, which does not have bins, in which the HPA model pick fruit and RTA models serves the HPA models by dispatching the fruit they picked and filled their bags.

4.4 Summary

The harvest process was modelled to simulate the current and proposed harvest and yield management methods based on a real apple orchard. The model consists of an orchard block, HPA models, RTA models, a tractor model, and a central controller to manage the harvest process. The block model is an undirected weighted graph which allows simulating multiple HPA and RTA models. The HPA and RTA models have independent attributes, but they collaboratively interact with each other. The model executes operation and optimization algorithms while collecting and analysing results.

Harvest management is time critical which includes hiring seasonal workers, harvesting fruit, and managing the yield. Moreover, it is an expensive process due to the use of extensive labour. Thus, it is difficult to conduct field experimentation during a harvest. Considering the cost and the time, creating a harvest model to simulate the harvest process is the best option.

The advantage of modelling the harvest is to test, analyse, and validate the proposed system's uncertainties. The developed models will provide a better understanding of the harvest behaviour and the system's response to changes. The results are visualized and analysed to explore better alternatives or to improve the optimization algorithms. Finally, the model allows viewing the systems from different perspectives by running multiple simulation scenarios.

CHAPTER 5: SIMULATING APPLE HARVEST, CURRENT, AND AUTOMATED YIELD MANAGEMENT METHODS.

5.1 Introduction

This chapter focuses on investigating the harvest and yield management by simulating the current method and the proposed automated method. The investigation of the current harvest and yield management method included simulating a scenario which employs three HPA models, who pick apples and fill fixed fruit bins, and a tractor model which transports full bins to a drop station. The investigation of the automated harvest and yield management method include simulating the proposed collaborative multi-agent system.

The proposed system simulation includes two scenarios which differ by employing either a single RTA or two RTAs. In the single RTA simulation scenario, the same three HPA models used in the tractor scenario were simulated here, but replaced the tractor model with an RTA model with the same speed and payload capacity. In the second scenario, the two RTAs replaced the tractor-equivalent RTA with two light RTAs of half the size and payload capacity. The two-RTA scenario was proposed to reduce the soil damage and service waiting time.

The simulation was a Monte Carlo simulation since the model has a deterministic principle. The simulation scenarios provided data to perform quantitative analysis and assess the system performance. The results analyse included fruit yield rate, service rate, and the RTAs serving time reliability. The RTAs serving time reliability analysis investigate the HPA models service waiting time. All simulation scenarios' results were compared to assess the system's practicality and validate its defined objectives.

The harvest and yield management process were modelled as discrete-events. In such a model, events occurred randomly and were triggered by the simulated models. In other words, the system model responds to changes that were driven by state changes at random time instants.

5.2 Modelling an Apple Orchard Block

Section 4.3.2 presented an orchard block model which has trees, drive rows, and a drop station. The model can create any fruit orchard's block based on its inputs which are the area, tree spacing, and row spacing. In this chapter, an apple orchard's block was created to provide a virtual environment to simulate the HPA, RTA, and tractor models. Each apple tree in the model produces random quantity of apple based on a uniform distribution as shown in Eq. (4.1).

5.2.1 Modelling an Apple Orchard Block for Current Harvest and Yield Management Method

Section 4.3.5 described the modelling of the current harvest and yield management method and discussed how to add bins to the block model. The block model inputs are 2m for tree spacing, 3m for row spacing, block's width of 31m, and block's length of 62m. The output is a block model with an area of 0.19ha, 352 trees, 176 bins, and a drop station. It is modelled as an undirected weighted graph with 528 vertices connected by 724 distance weighted edges.

5.2.2 Modelling an Apple Orchard Block for Automated Harvest and Yield Management Method

The proposed automated method removes the bins and replaces the tractor with RTAs. Thus the block model is similar to the above model but without fruit bins. Both models utilise the same block/tree geometry. In both block models, the drop station vertex is located at point (8m, -20m) which is 21.5m away from the first tree (*tree0*) and 98.7m away from the last tree (*tree351*). These distances were the shortest calculated paths from the drop station to *tree0* and *tree351*.

5.3 Simulating HPA for Picking Fruit

The purpose of this simulation scenario is to show how the HPA models, which were presented in Section 4.3.3, behave while picking fruit and to demonstrate the outcome distributions of

the predefined picking experience parameters. The HPA models should have the same behaviour in all of the simulated scenarios. Thus, this simulation does not discuss if the picked fruit are dispatched by the RTA models or if the HPA models walk with them to fill bins.

5.3.1 HPA Model Simulation Setup

An HPA model was assumed to have three levels of experience which are a beginner, moderately skilled, and highly skilled which were determined based on the average fruit picking cycle time. The average fruit picking cycle time was calculated by Eq. (4.3). Each apple picking cycle time random variable was generated from a uniform distribution as indicated in Eq. (4.4). The above values were obtained from the following calculations:

A. Beginner picker's experience:

   For 8h working day, a beginner HPA can produce 2 tons of apples, where a single apple weighs 90-150g

   Number of 150g apples produced per day $= \frac{2000000\ g}{150\ g} = 13333$

   Time to pick a single 150g apple $= \frac{8\ h * 60\ \text{min} * 60\ s}{13333} = 2.1s$

   Number of 90g apples produced per day $= 22222$

   Time to pick a single 90g apple $= 1.2\ s$

   Beginner HPA's picking single apple cycle time base on Eq. (4.4) is

   $E \sim U([1.2, 2.1])$

B. Moderately skilled picker's experience:

   A Moderately skilled HPA can make 3 tons of apples per 8 h working day

   Number of 150g apples produced per day $= 20000$

Time to pick a single 150g apple $= 1.44\text{s}$

Number of 90g apples produced per day $= 33333$

Time to pick a single 90g apple $= 0.86\text{s}$

Moderately skilled HPA's single apple picking cycle time $E \sim U([0.86, 1.44])$

C. Highly skilled picker's experience:

A highly skilled HPA can make 4 tons of apples per 8 h working day

Number of 150g apples picked per day $= 26667$

Time to pick a single 150g apple $= 1.1\text{s}$

Number of 90g apples picked per day $= 44444$

Time to pick a single 90g apple $= 0.65\text{s}$

Highly skilled HPA's single apple picking cycle time $E \sim U([0.65, 1.1])$

Three HPAs models were created to simulate to pick apples from the block model shown in Fig.5.2. The first HPA's ID is HPA1 and is a beginner picker. The second HPA's ID is HPA2 and is a moderately skilled picking experience. The third HPA's ID is HPA3 and is a highly skilled picker. The model generates a random variable speed for each picker from a uniform distribution specified for each level of experience. The threshold for every HPA in this simulation is set to 20kg.

5.3.2 HPA Model Simulation Results

The simulation results for the three simulated HPAs' models are shown in Fig.5.1. The results show three normally distributed sets of data representing the duration each HPA took to fill a 20kg picking bag. The three data sets are normally distributed since the Central Limit Theorem

(CLT) stated that the randomly and independent observations from predefined distributions will have a normal distribution. In this scenario, each tree produced 18-19 picking bags. The results are presented also in Table 5.1.

Table 5.1. HPA models picking apple simulation results showing picking bags filling time.

HPA's ID	Picking Skills Level	Bag's weight in kg	No. filled bags	Total weight of produced apples in tons	No. visited trees	Min. bag's filing time in s	Max. bag's filing time in s	Mean bag's filing time in s	Standard deviation in s
HPA1	Beginner	20	1479	29.6	82	262.4	293.4	275.7	4.6
HPA2	Moderately skilled	20	2100	42	117	183.6	203.1	192.3	3.1
HPA3	Highly skilled	20	2745	54.9	153	138.6	154.4	146.2	2.3

The HPA models who has better experience are more efficient than the ones with less experience. HPA3 was twice as fast as HPA1 and therefore produced double the quantity. The model represents apple harvest and pickers efficiency in the real world as defined by the model. In reality, highly skilled pickers move faster and produce more bins per day. Discussions with the apple growers revealed that the highly skilled pickers have efficient picking strategies such as starting to pick apples from the bottom of the tree all the way to the top, picking furthest apples to nearest, and moving effectively. The resulted distribution of bag filling time is normally distributed.

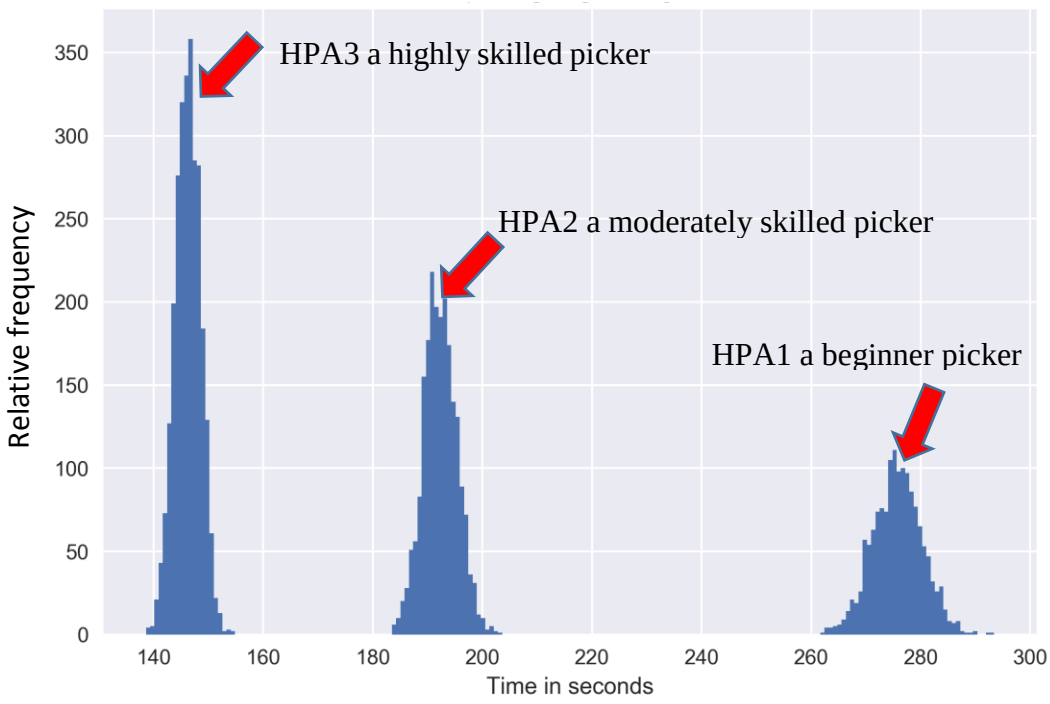


Fig. 5.1. The HPA models picking bags' filling time frequency distribution left to right HPA3, HPA2, and HPA1.

5.4 Simulating the Current Harvest and Yield Management Method (Tractor Model)

The current yield management method employs orchard tractors to transport empty bins to the orchard drive rows. The usage of tractors in yield management is discussed in Section 2.2.2 while a tractor simulation model is discussed in Section 4.3.5.

The current harvest and yield management method sequence diagram which represents the simulation discrete events is shown in Fig.5.2. The process starts by the having HPAs walk to trees to pick apples and fill their picking bags. Once a bag is full, the HPA walks to a bin placed within the drive row to empty the bag. The HPA returns to pick apples and repeat the process until the tree is wholly harvested and moves to the next tree. A supervisor calls a tractor to transport full bins to the drop station. The simulation stops when all the trees in the orchard's block are entirely picked.

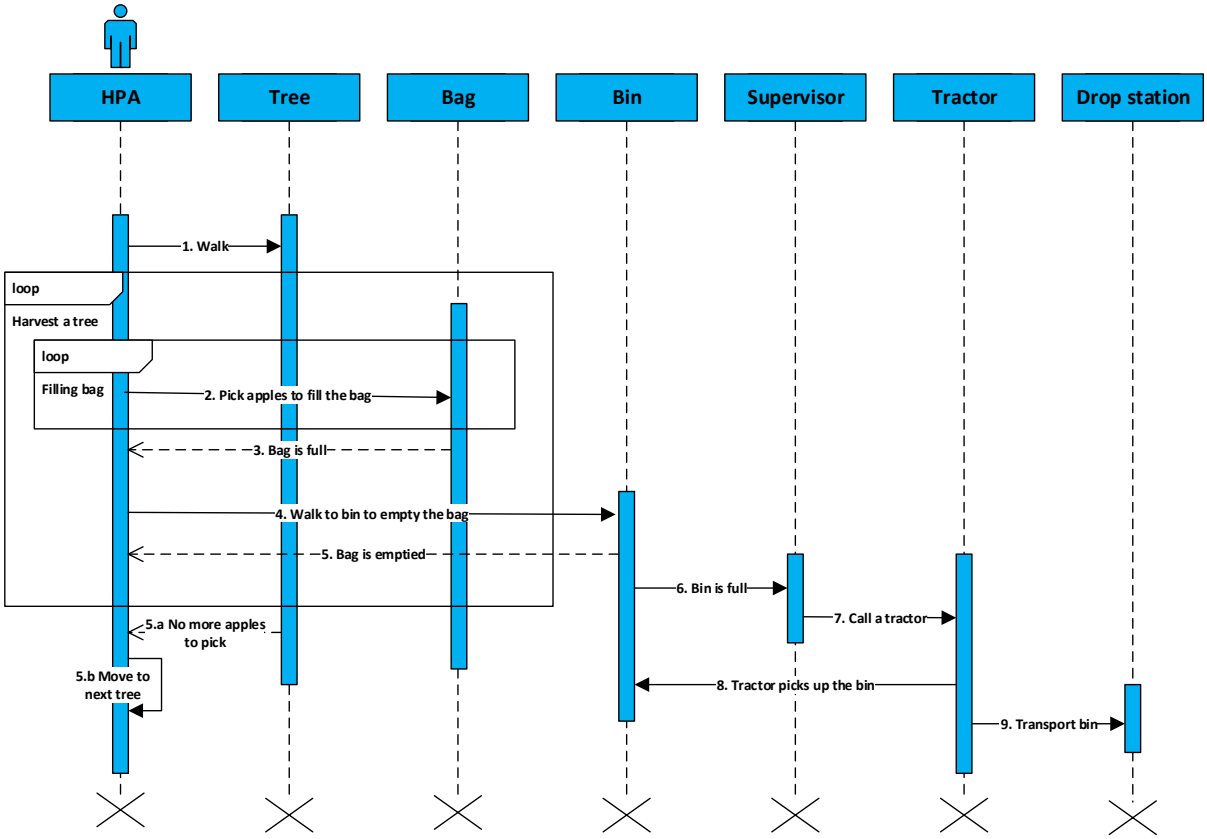


Fig. 5.2. Sequence diagram for apple harvest as a discrete-event simulation employing a tractor.

5.4.1 Tractor Model Simulation Setup

In this simulation scenario, the same three HPA models were simulated. They generated the same results as shown in Fig.5.1 since they had the same experience in both scenarios. Moreover, both scenarios had the same random generator seed which generated the same sequence of random variables. The tractor model has a payload of 800kg and a speed of 2 m/s. The tractor is assumed to have a constant speed and travel smoothly in an ideal collision-free path. The HPA models walking speed was assumed to be 1 m/s. The results are the observation of the HPAs walking time to bins and bins' transporting time by a tractor.

5.4.2 Tractor Model Simulation Results

In this simulation, the tractor made 176 round trips to dispatch full bins from the drive rows and transport them back to the drops station. The tractor model was assumed to be moving in

ideal collision-free environment. The tractor's round trip took 21.8-91.5s with a mean of 54.8s. The tractor took 160.76 min to pick up and transport 176 bins. The sum of the total HPA model walking time to a single bin was 64.9-68.5s with a mean of 64.9s. Table 5.2 shows the total time each HPA took to walk to the fixed bins to empty their picking bag.

Table 5.2. HPA models total walking time to fill bins and total number of filled bins.

HPA's ID	Picking skills level	Bag's weight in kg	No. of bins filled	No. visited trees	Total walking time in min
HPA1	Beginner	20	41	82	88.98
HPA2	Moderately skilled	20	58.5	117	127.14
HPA3	Highly skilled	20	67.5	153	164.94

The total unproductive walking time was 381.06min. The three HPA models took a total time of 116.97h which is 14.6 regular 8h working days to harvest the block model. The total non-harvesting unproductive time which is the HPA models walking time to bins and tractor operating time was 9.03h which made around 7.72% of the total harvest time. This is considering the best condition where the HPA models were continuously picking and the tractor model was moving in ideal conditions without stopping.

5.5 RTA Model Simulation Setup

5.5.1 Arrival Rate of Dispatching Requests of and Service Rate Setup

The HPA models fill their picking bags and send dispatch requests to the central controller. The arrival of these dispatching requests occurrences are independent. The duration between the consecutive arrivals of two dispatching requests is observed and the inter arrival of dispatching requests $\mu_b[min]$ is the arithmetic mean of these observed durations. The mean of dispatching requests arrival rate λ is the constant average rate at which dispatching requests

arrive within a unit time $t_b[min]$. It is expressed as the number of requests arriving within one minute as shown in Eq. (5.1).

$$\lambda = \frac{1}{\mu_b} \tag{5.1}$$

The arrival of dispatching requests is assumed to follow a Poisson distribution. The requests arrival probability distribution can be estimated by the Poisson's probability mass function (PMF) as seen in Eq. (5.2). Let the $P(k)$ be the probability of observing a k number of dispatching requests in a specific unit time with the mean arrival rate of λ.

$$P(k) = \frac{\lambda^{k}}{k!} e^{(-\lambda)} \tag{5.2}$$

The HPA models have to wait for RTA model to dispatch their picking bags. The HPA models service waiting time was also observed to calculate the whole system service rate and to analyse the RTAs serving time reliability. The constant average service rate α per unit time $t_s[min]$ is expressed as the time taken between serving a number of requests within one minute as shown in Eq. (5.3) where $\mu_s[min]$ is the arithmetic mean of the observed service waiting times in which the HPA models had to wait for the RTA models to serve them.

$$\alpha = \frac{1}{\mu_s} \tag{5.3}$$

Each request serving time is continuous and independent and assumed to follow an Exponential distribution. The probability distribution of the service time per minute is determined based on the exponential probability density function (PDF) as seen in Eq. (5.4). Where $P(s)$ is the probability of observing s number of requests processed in a unit of time with α.

$$P(s) = \begin{cases} \alpha\, e^{-\alpha\, s} & s \geq 0 \\ 0 & s < 0 \end{cases} \tag{5.4}$$

The λ and α values are used as inputs for a queueing model which is will be discussed in Chapter 6. The queueing analysis will provide a better understanding of the system service time and RTAs assigning mechanism to determine the optimal service waiting time and the optimal number of RTAs required. Fig. 5.3 shows the arrival of dispatching requests which are waiting to be served by the RTAs which are being assigned by the system.

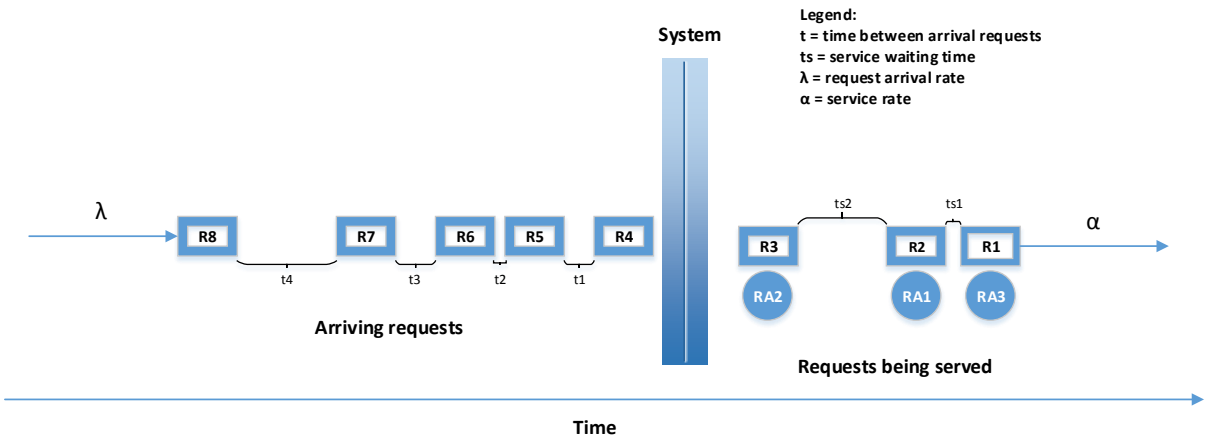


Fig. 5.3. The arrival and serving mechanism of dispatching requests by the system.

There is a difference between the Poisson arrival of the dispatching requests and the Exponential serving of these requests. The difference is that the Poisson distribution is a discrete distribution which models the probability of having a number of requests in a given period of time while the Exponential distribution is a continuous distribution which models the probability of the waiting time in which the requests are being served.

5.5.2 Monte Carlo Method Setup

The Monte Carlo method was used to provide a good approximation of the dispatching requests inter-arrival and the mean of the service waiting time for the single RTA scenario, two-RTA scenario, and multi-RTA scenarios. Each scenario will be simulated 10000 times while a Monte

Carlo convergence analysis will be conducted in Section 6.5.4 to determine how many simulations are needed to achieve the desired accuracy.

For Each simulation, a unique pseudo-random generator seeding value generates a unique random sequence, and the results are appended to a global data frame. At the end of the Monte Carlo simulation, the algorithm averages the requests inter-arrival values and the service waiting time values for every simulation and then takes their average.

5.5.3 Service Time Reliability Analysis Setup

The primary objective of analysing the RTAs serving time reliability is to access and control HPA models service waiting time. The model analysed the picking bag dispatching time and the HPAs' service waiting time as these were deemed to be determinants of the RTAs serving time reliability. The dispatching request made by an HPA is the start of an event which ends when an RTA arrives at the request location. The duration between the event's start and end is unambiguous and well-defined. All events occurred during the simulation, and none was censored. The RTAs serving time reliability analysis provide good estimates about the proportion of requests handled over time and the effectiveness of improving the system algorithms which will be discussed in Chapter 6.

The RTA models service time reliability is determined by analysing the expected time to dispatch a picking bag which is the HPA models service waiting time. In other words, the observed event is the duration between when an HPA places a dispatching request until the assigned RTA dispatched the yield.

Let *T* be a non-negative continuous random variable representing the service waiting time with cumulative distribution function $F(t)$ on the interval $[0, \infty)$. The $R(t)$ is the successful dispatching function as given in Eq. (5.5). The $R(t)$ gives the probability of service waiting time which has exceeded time *t*.

$$R(t) = Pr\{T > t\} = 1 - F(t) = \int_t^{\infty} f(x)dx, \qquad where$$
$$f(x) = \frac{d}{dx}F(x) \tag{5.5}$$

The results were fitted to the Kaplan-Meier estimator which estimates the reliability function $R(t)$ Eq. (5.6). Let $\hat{R}(t)$ be the estimate of the reliability function when at least one event happens, d_i is the number of events at t_i, and t_i is the number of events that have not occurred yet. The estimator confidence interval is an Exponential Greenwood confidence interval.

$$\hat{R}(t) = \prod_{i:\, ti \le t} \left(1 - \frac{di}{ni}\right) \tag{5.6}$$

5.6 Simulating Harvest and Yield Management Automation (Single Big RTA Model)

5.6.1 Single RTA Model Simulation Set up

In this simulation scenario, a single RTA replaced the tractor, cut out supervisor role, and the use for bins. The working concept of the proposed harvest and yield management automation method and the multi-agent system is discussed in Section 3.2.1. The RTA model discussed in Section 4.3.4 is simulated as a mobile bin to dispatch fruit bags from HPAs. The discrete-event simulation's sequence diagram is shown in Fig. 5.4. The RTA simulation scenario has the same three HPA models which generated the results shown in Fig. 5.1. This simulation is a Monte Carlo simulation in which its setup as shown in Section 5.6.1. The service reliability analysis Section 5.6.2 was also applied in this simulation.

The simulation starts by the three HPA models walking to trees to pick apples and fill their picking bag. The central controller responds to the dispatching request and assigns a suitable RTA. In this case, there is only a single RTA operating. The HPAs resume picking once the RTA dispatches their bag. The RTA serves several HPAs and delivers the yield back to the

drop station. The simulation stops when all the trees are entirely picked. The RTA is considered in this simulation as an independent resource and is assigned by the central controller based on the first-in-first-out (FIFO) queueing method. The RTA has a maximum payload of 800kg and a constant speed of 2 m/s. The RTA is assumed to be moving in an ideal collision-free path with no obstacles or disturbances.

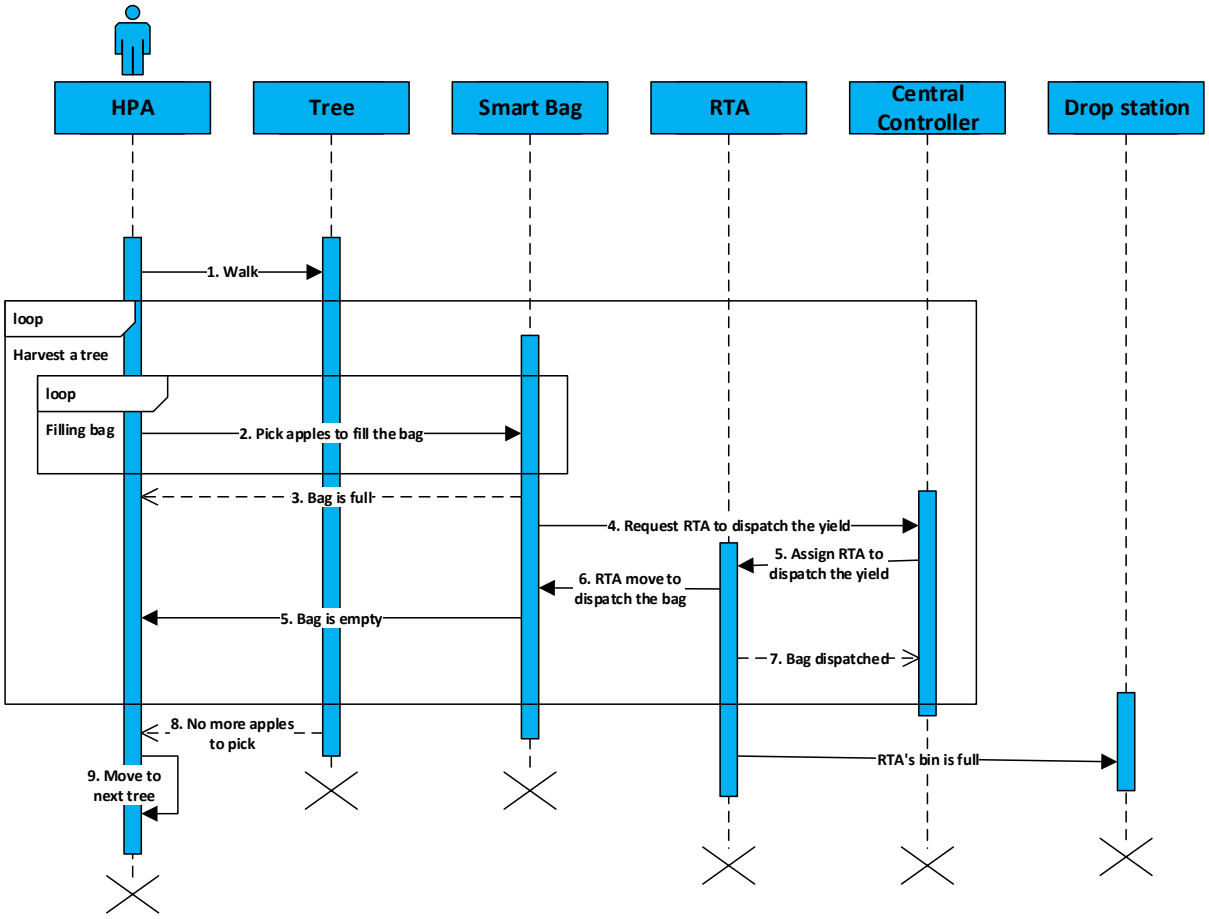


Fig. 5.4. Sequence diagram for apple harvest as a discrete-event simulation model employing a RTA.

5.6.2 Single RTA Model Simulation Results

The time difference between the arrivals of the picking bags or requests made by all the HPA models during the simulation was observed as shown in Fig.5.5 and the mean of dispatching requests arrival rate λ was calculated by Eq. (5.1). The total number of dispatching requests made by the HPA models when filling picking bags in this simulation was 6324. The duration

between the occurrences of two dispatching requests was 0.04-325.9s with a mean of 65.3s. Thus, the dispatching requests inter arrival μ_b is 65.3s.

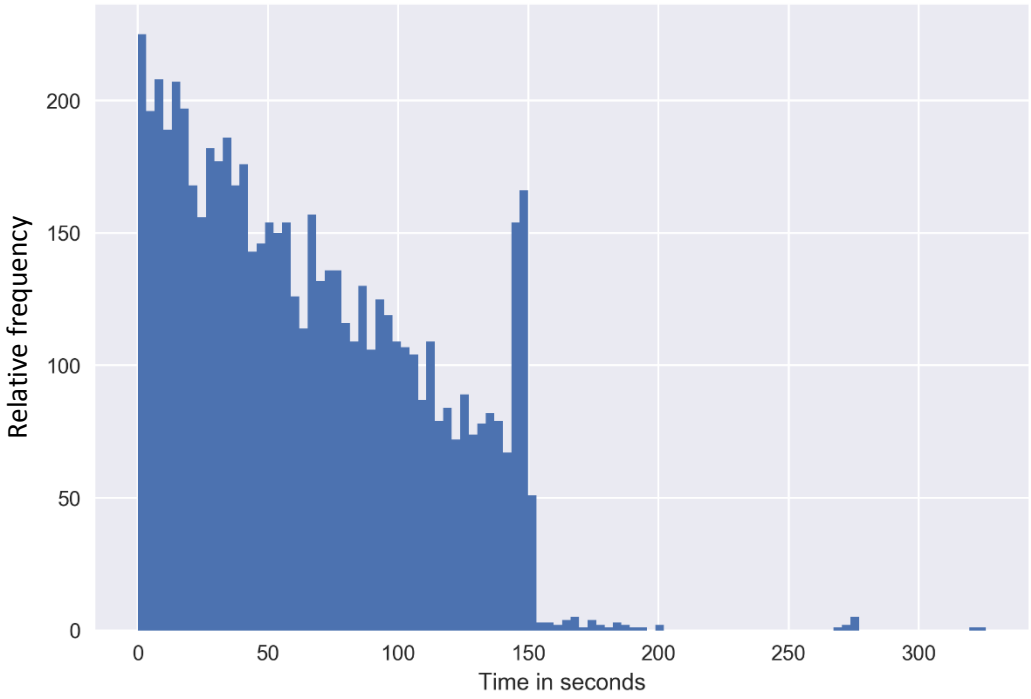


Fig. 5.5. The frequency distribution of the time difference between the arrivals of two dispatching requests.

The arrival of dispatching requests had a mean arrival rate λ of 0.92 per 1min or 4.6 per 5min as calculated by Eq. (5.1). The system approximately receives a dispatching request every minute. The mean arrival rate λ is a unitless quantity which represents the expected number of requests arriving within a period of time. The requests inter-arrival probability distribution is assumed to follow a Poisson distribution which is demonstrated in in Fig.5.6 by taking the λ per min and the total number of observations k as parameters to Eq. (5.2). The request inter-arrival probability is assumed to have a Poisson distribution since the λ is used in Chapter 6 in a queueing analysis to investigate the system response while improving its service time.

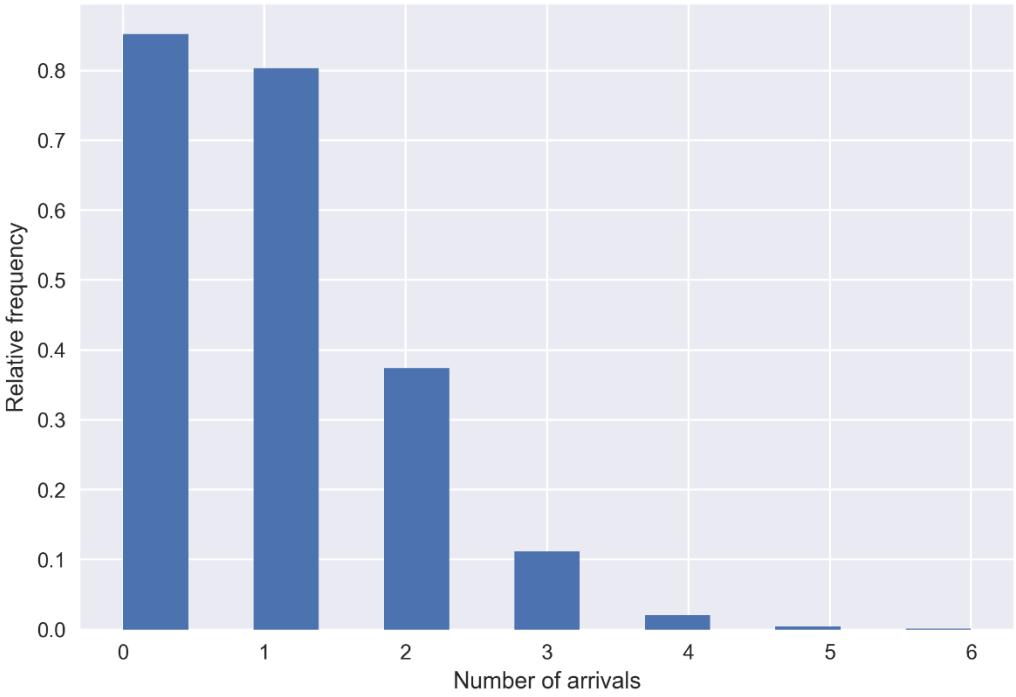


Fig. 5.6. The Poisson probability distribution of the requests arrival rate per minute.

The frequency distribution of all HPA's service waiting time is shown in Fig. 5.7, while the waiting time distributions of HPA1, HPA2, and HPA3 are presented in Appendix B1. Table 5.3 shows each HPA model service waiting time.

Table 5.3. The HPA models service waiting time results.

HPA's ID	Picking skills level	Bag's weight in kg	No. of dispatching requests made	Min. service waiting time in s	Max. service waiting time in s	Mean of service waiting time in s
HPA1	Beginner	20	1479	0	76.6s	3.4
HPA2	Moderately skilled	20	2100	0	76.2	2.9
HPA3	Highly skilled	20	2745	0	82.4	2.9
All	-	20	6324	0	82.4	3

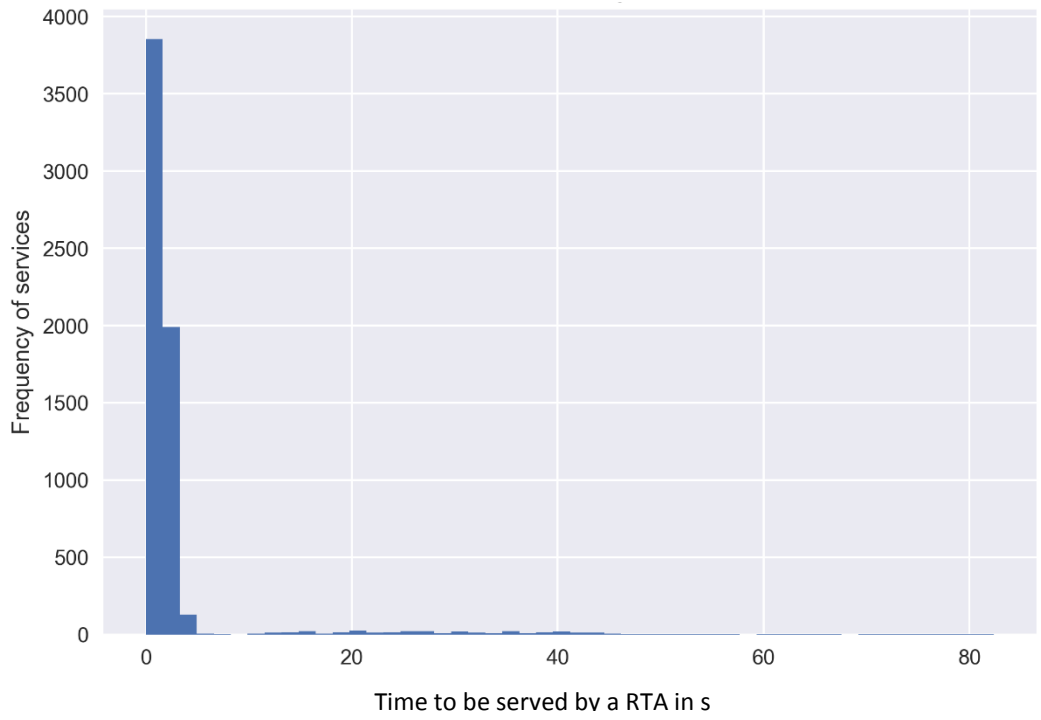


Fig. 5.7. HPA models service waiting time frequency distribution.

The mean of the service time is 3s and thus, the number of requests to be served within 60s is 20 as calculated by Eq. (5.3). The constant average service rate α = 20 per min. The value of α is unitless quantity which represents that the system is expected to served 20 dispatching requests per 1 min. The Exponential probability distribution of the service time per a single minute is shown in Fig. 5.8 which demonstrate the assumption that the system service time is exponentially distributed over time which will be used later in Chapter 6.

Based on the Monte Carlo simulation method, the dispatching request inter arrival was 65.1s, and the mean of service waiting time was 3s. Thus, the arrival rate of dispatching requests λ which were expected to be received by the system within 1 min was estimated as 0.92, and the constant average of service rate α as 20.

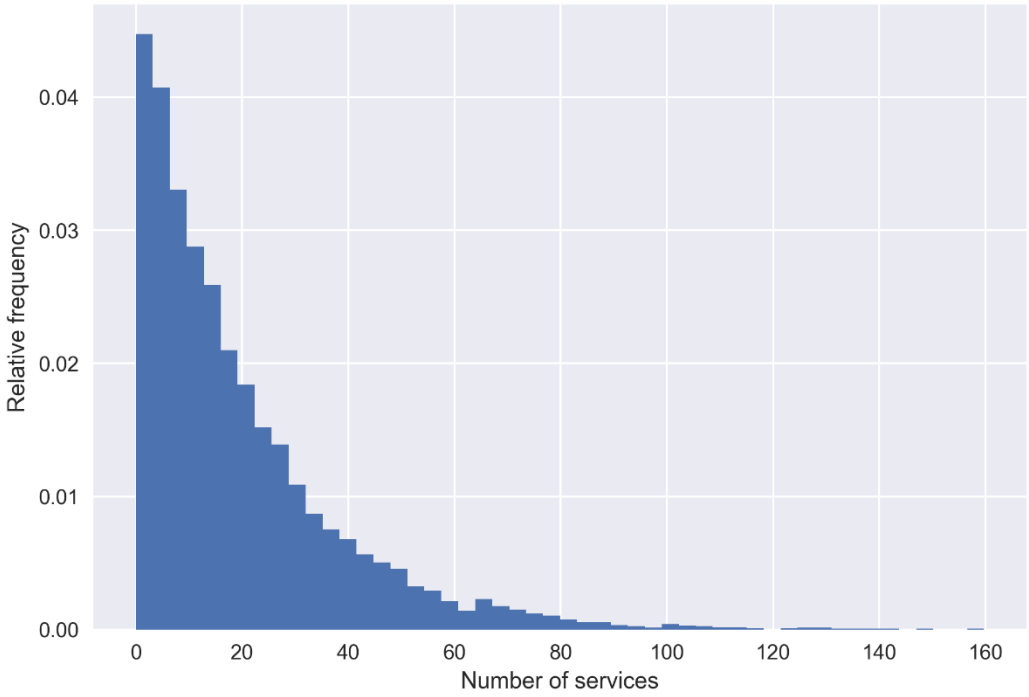


Fig. 5.8. An exponential distribution showing the system expected compilation of services per minute.

The Kaplan-Meier plot estimating the system service time reliability function is shown in Fig. 5.9. The plot shows the waiting time for all the HPAs serviced by a single RTA. The estimated $\hat{R}(t)$ = 0.951 at t = *0*s since 278 requests out of 6323 were dispatched instantly. There is a 4.9% chance that the request is handled instantly. Moreover, $\hat{R}(t) = 0.41$ at t = 1 which shows there is a 59% chance that a request is handled with 1s since the HPA models are moving together and the RTA is waiting nearby. The Kaplan-Meier plot has a steep decline at the time interval 0-5s. However, it smoothly declines from 5s onward. The system is quite reliable since there is only 5.3% chance that a request might take more than 10s to be handled. The reliability function estimator plot for each HPA is shown in Appendix B2. The blury double lines along estimator plot represents the confidence interval which was calculated by using the exponential Greenwood method.

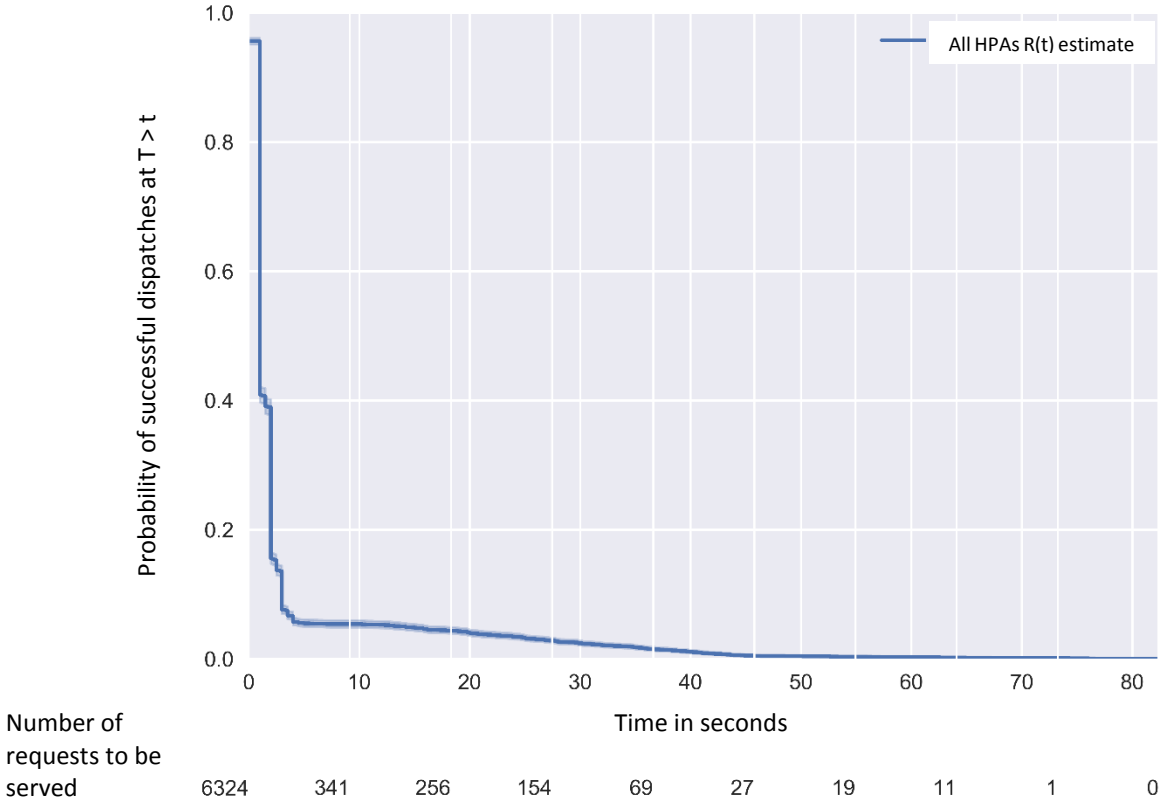


Fig. 5.9. Kaplan-Meier plot for the system with a single RTA successful dispatching function. The first line on the grid represents the number of unserved dispatching requests shown under the time x-axis.

The total harvest time was 114.7h which is about 14.3 working days. The total service waiting time was 5.3h which made 4.6% of the total harvest time. The non-harvesting unproductive time was improved by 41.3% when replacing the tractor with an RTA. The System optimizes the cost by cutting out supervisors, bins, tractors, and tractor drivers. The simulated RTA has the same capacity as a normal tractor which is 800kg. Heavy agricultural vehicles cause soil damage and compaction. It is more efficient and soil friendly to deploy multiple RTAs which each have a small payload capacity.

5.7 Simulating Harvest and Yield Management Automation (Two Small RTA Models)

This simulation scenario investigated the simulation of two light RTA models to reduce service waiting time and soil damage. The design concept of the proposed collaborative multi-agent system is to employ multiple RTAs to cooperate with and serve multiple HPAs. The literature

review concluded that multiple simple robots are flexible; can handle complicated tasks; cheap; and more efficient than a single advanced robot.

5.7.1 Two RTA Models Simulation Setup

The two simulated RTA models were homogenous which had 400 kg payload capacity and a speed of 2 m/s. The same three HPA models which were simulated in the previous scenarios were also simulated in this scenario. The HPA models were expected to behave similarly to previous scenarios since their pseudo-random number generator was seeded with the same value to generate the same sequence of random numbers. However, as soon as the first request is dispatched the HPAs interacts with the RTAs the simulation is no longer the same as the single RTA. The central controller assigned the RTA models based on who became available first would serve the first coming request.

The Monte Carlo simulation method was also used. The two RTA models were simulated 10000 times with 10000 pseudo-random generator different seeding values. On each simulation, the seed was set, and the results were appended to a global data frame. The algorithm averaged the average of each simulation results at the end of the Monte Carlo simulation cycle.

The RTA model reliability was determined by analysing the expected time to dispatch a picking bag which is the HPA's service waiting time. Thus, the system reliability was estimated by fitting the service waiting time to the Kaplan-Meier estimator Eq. (5.6).

The log-rank test quantitatively compared the difference between the two scenarios' reliability analysis. The log-rank test was chosen since the observed data were the service's waiting time, and they are highly skewed throughout the whole observation. The null hypothesis *H0* assumed

that there was no difference if the system deploys two smaller RTAs or one large RTA. The log-rang test p-value less than 0.01 was considered to be statistically significant.

5.7.2 Two RTA Models Simulation Results

The results showed that HPA3 made 2779 requests and waited for an RTA 0-45.9s with a mean of 3.5s while HPA2 made 2112 requests and waited for 0-46.9s with a mean of 4s. In the meantime, HPA1 also made 1479 requests and waited for 0-44.8s with a mean of 4.1s. The total numbers of dispatching requests made by all HPA model was 6370 and the service waiting was 0-46.9s with a mean of 3.8s. The service waiting time frequency distribution and bags arrival time difference frequency distribution for all the HPA models are shown in Appendix B3.

The number of dispatching requests made by HPA models in this scenario was larger than the number of requests made in the single RTA model scenario. This change is caused by the way the HPA and RTA models interacted which resulted in different services time. In other words, the HPA models has to wait different amount of time to be served. For example, HPA3 could start picking a new bag sooner and therefor pick more, faster. As an additional example, in the single RTA model scenario, HPA3 would finish picking tree30 and move to tree32 while in the multiple RTA models scenario the HPA3 would finish picking tree31 and move to tree33.

The resulted dispatching requests mean arrival rate λ per min in this simulation is 0.92 which has a similar Poisson distribution to the single RTA model scenario as shown in Fig. (5.6). Moreover, the service constant average rate per min α is 15.8 which is less than the single RTA model scenario's α value. The α value in this scenario is less since the RTA models had smaller payload capacity and made more trips to the drop station. The following chapters discuss how to improve the service waiting time when deploying multiple small RTAs. The system service rate probability distribution is shown in Appendix B3.

Based on the Monte Carlo method, the mean of the bags' occurrences time difference was 65.4s, and the mean of the service waiting time was 3.8s. The constant average arrival rate λ was 0.92 per min, and the constant average service rate α was 15.8 per min.

The Kaplan-Meier plot estimating the system's reliability function is shown in Fig. 5.10. The estimated $\hat{R}(t)$ = 0.77 at t = *0*s meaning that there is a 23.2% chance a request is handled instantly since 1477 requests out of 6370 were dispatched at 0s waiting time. Moreover, $\hat{R}(t)$ = 0.39 at t = 1s which shows that there is a chance of 61% that a request is handled within 1s. The Kaplan-Meier plot had a steep decline at the time interval 0-5s. However, it smoothly declined from 5s onward. The system is quite reliable since only 4.5% of the total number of requests took more than 5s to be handled. The reliability function estimator for each HPA's estimator plot is shown in Appendix B4.

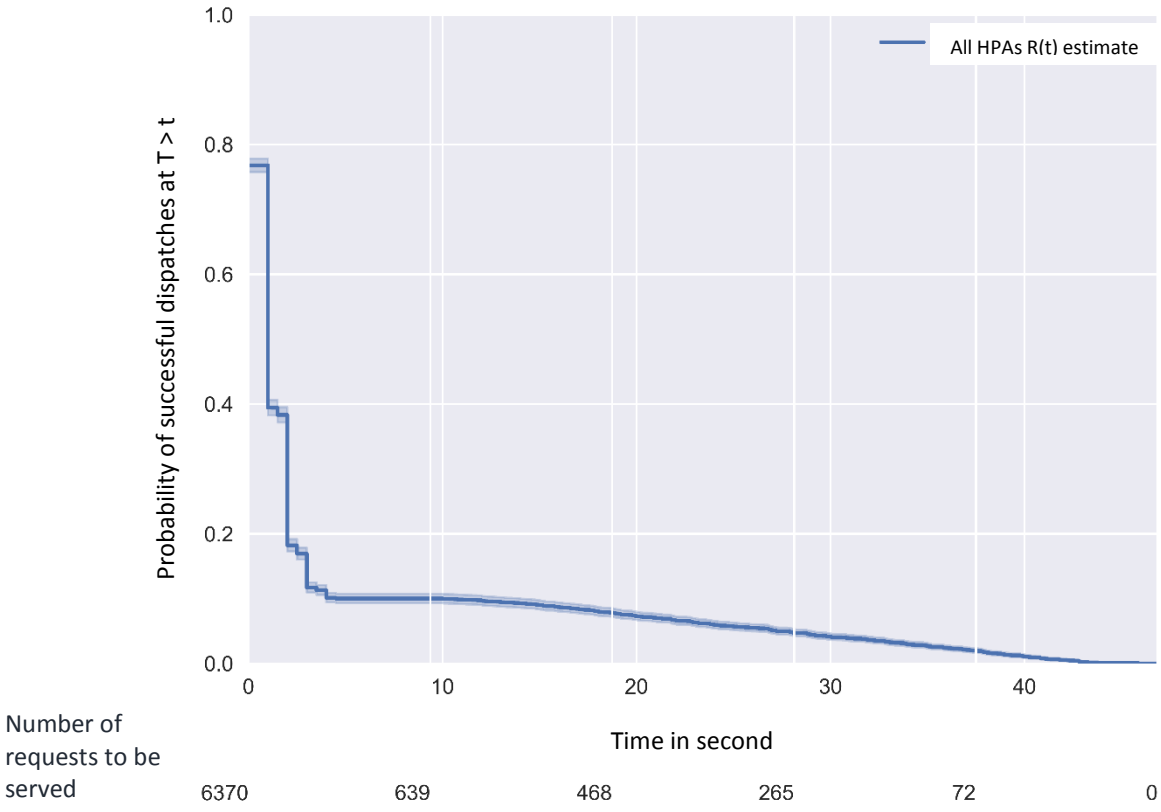


Fig. 5.10. Kaplan-Meier estimator plot for the system successful requests dispatching function while using two RTAs. The first vertical line on the grid represents the number of unserved dispatching requests shown under the time x-axis.

There is a noticeable difference when comparing the reliability analyses of the single 800kg RTA model and two 400kg RTA models as shown in Fig. 5.11. The log-rank test was applied to compare the difference between the two reliability estimate distributions. The result of the log-rank test with a confidence value $\alpha = 0.99$, a degree of freedom $df = 1$, and Chi-squared null distribution rejects the null hypothesis with a test statistic value of 86.033 and the p-value < 0.00001.

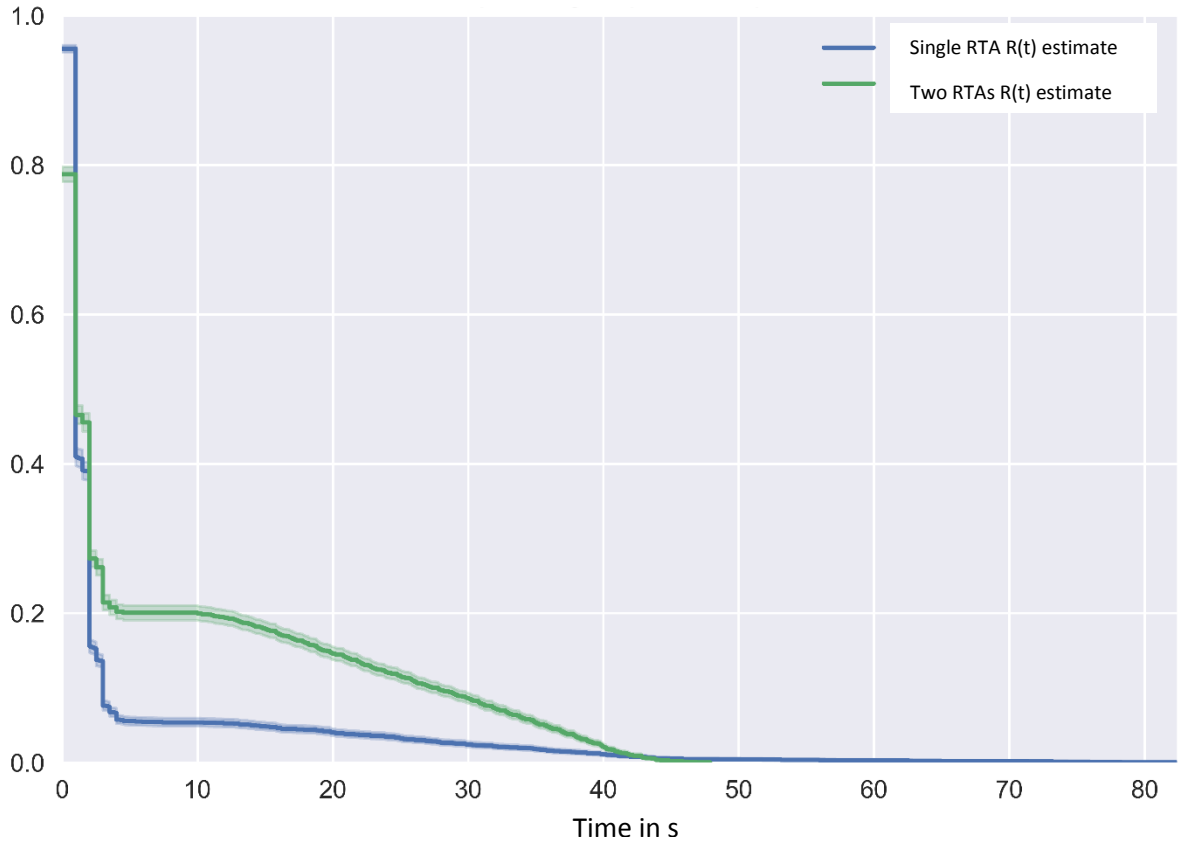


Fig. 5.11. The comparison service time reliability analysis for employing a single RTA and two RTAs.

In this scenario, the total harvest time is 115.6 hours which is about 14.4 working days and the total service waiting time was 6.7h which improved the total unproductive non-harvesting time by 25.8% when compared to the tractor model simulation scenario. The non-harvesting unproductive service waiting time contributes to 5.7% of the harvest time. However, the maximum service waiting time was 46.9s which was improved by 43.1% when compared to

the single RTA model. The number of requests which were dispatched instantly increased from 278-1477 which was improved by 36.9%. The use of multiple light RTAs improved the system efficiency and minimized soil damage.

However, the system took longer total service waiting due to the way the central controller selects and assigns the RTAs. Currently, the central controller assigns the RTAs the first available RTAs to serve dispatching requests and neglects their locations. In many cases the non-assigned RTA is closer to the request than the assigned RTA. The service waiting time optimization is investigated in Chapter 6 by developing smarter RTAs selections algorithm.

5.8 Summary

The discrete-event model developed in Chapter 4 was simulated to investigate the conventional and the automated harvest and yield management methods. The simulation were examined by using the Monte Carlo analysis. The simulation investigated both methods behaviour during apple harvest including HPA models, RTA models, and a tractor model. The results analysis included fruit picking rate, service rate, and RTA models' reliability.

The current method was simulated and investigated by launching three HPA models to pick apples and fill fixed fruit bins which were transported by a tractor model. On the other hand, the automated method was simulated and investigated by simulating the same three HAs to pick apples who were served by a single RTA model. The single RTA replaced the tractor model and had the same payload capacity and speed.

The automated method improved the non-harvesting unproductive time by 41.3% and optimized the cost by cutting out the need to hire supervisors and tractor drivers. Automating the supervisor and tractor driver will reduce the annual harvest cost by 58-63%.

The replacement of a tractor model with an equivalent RTA model resulted in long service waiting time and did not resolve the soil damage. Thus, another scenario simulated two small

RTA models in which each RTA model has half the single RTA model's payload capacity. This scenario improved the maximum service waiting by 43.1% and increased the instantly served requests from 278 to 1477. However, having a small payload capacity resulted in more returns to the drops station and hence prolonged the total service waiting time.

The automation of harvest and yield management automation method has optimized the harvest time and enhanced the pickers' efficiency. Moreover, it has improved the process safety at a minimum cost. It improved the safety by cutting out HPAs' repetitive walk to the fruit bins, minimized the tripping hazards, and limited carrying overweight loads. It optimized the cost by discarding the use of fruit bins, tractors, tractors divers, and pickers' supervisors.

The total service waiting time was longer when employing two small RTA models due to their frequent returns to the drop station. The system response time, the optimal number of required RTAs, and the RTA's selection method will be investigated in the following chapter by improving the control system RTA's assigning algorithm.

However, this model did not take into account the HPAs random movement while moving and only consider having a pre-defined path to be followed. The HPAs actual movement while picking was not considered such as climbing ladders, and if they decide to temporary leave the block and later resume picking from a different tree in a different row. The current HPA model is very simple and generic and assumes perfect movement. This need to be considered in the future to improve the simulation results.

CHAPTER 6: DEVELOPING DESPATCHING ALGORITHMS TO OPTIMIZE THE SERVICE WAITING TIME, NUMBER OF SERVING ROBOTIC TRANSPORTING AGENTS, AND SOIL DAMAGE.

6.1 Introduction

Chapter 5 verified the collaborative automation solution's feasibility in a series of simulations. However, the automation method did not solve the soil compaction problem and resulted in long service waiting times. This chapter investigates the improvement of the RTAs selection and request dispatching algorithms.

In particular, this chapter investigates the development of smart dispatching algorithms to reduce the service waiting times, the RTA's payload capacity, and the number of RTAs required. These algorithms are as follows

1. The First Available First Serves (FAFS) dispatching algorithm
2. The Dynamic Distance (DD) dispatching algorithm for homogenous RTAs
3. The Dynamic Distance and Best fit (DDB) dispatching algorithm for heterogeneous RTAs.

As concluded in Chapter 2 that using multiple heterogeneous robots improves the automation of tasks in dynamic agricultural environments. Moreover, the versatile ability of each robot improves the system response to the HPAs random and unpredictable behaviour, provided that a smart dispatching algorithm is used.

The algorithms developed were simulated using the methods described in Chapter 5. All simulation scenario results were compared to assess algorithm performance. The analysis included dispatching requests arrival rate, service rate, and the RTAs service time reliability. The queueing theory and analysis were introduced to study the algorithms responses and

optimize their outputs. The queueing analysis provide a quantitative approach to assess the developed algorithm and the system performance.

6.1.1 Background

The use of bigger and heavier agricultural vehicles to quickly perform large-scale agricultural operations is a current trend [3] [5]. These vehicles can combine implements (devices attached to the vehicles to perform specific tasks) which increase their weight. The use of heavy agricultural equipment results in soil compaction which gets deeper as the weight of the vehicle increases [4] [5]. Compaction affects the soil quality by reducing the water infiltration rate. Such soil limits the penetration of water to the subsoil, lessening root spread, and increases the surface water ponding [87].

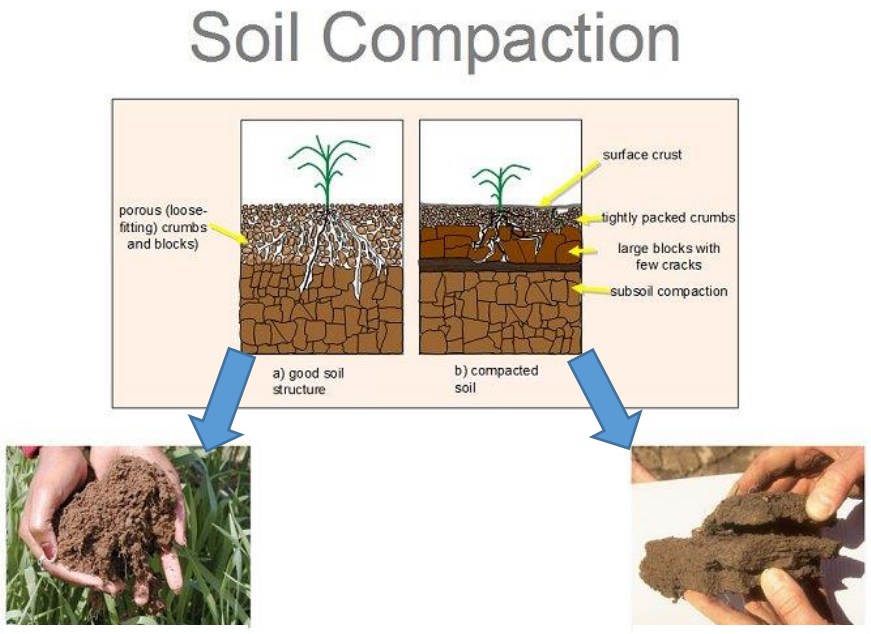


Fig. 6.1. Soil compaction effect on roots development [88].

Soil compaction affects plant growth as shown in Fig. 6.1. The use of tractors is the leading cause of soil compaction due to their weight. The tractors combine the ground contact pressure with the axle load to provide high torque traction [3]. They cause the same soil damage if used

for heavy agricultural tasks such as soil tillage or for light tasks such as transporting. Thus, a tractor is not an ideal choice to transport yield.

One possible effect of using heavy agricultural equipment is shown in Fig. 6.2. The figure shows a tractor sinking in the mud and causing significate damage. In such a scenario, when agricultural equipment gets stuck in the mud, the driver, the soil, the farmer, or the weather are to be blamed. In fact, the real case is the tractor itself since it is heavy and has a tracking mechanism which forces the tires to sink in any soft or wet soil.

Fig. 6.2. A tractor stuck in an orchard causing significant soil damage [6].

The treatment of compacted and damaged soil is an extra farming cost. However, framers typically assume that soil tillage and cultivation take would care of the compaction, but they only treat the topsoil compaction while the subsoil compaction remains untreated [88]. The use of heavy agriculture equipment is the primary cause of subsoil compaction.

The best way to treat soil compaction is to control its causes by reducing vehicle weight and increasing the vehicles speed [4]. Deploying a swarm of lightweight mobile robots is a suitable

alternative to heavy tractors. Another advantage of using a swarm system is that each robot can manage tasks individually and collaboratively [88]. The future objective of agricultural robots is to create a team of smart machinery to maintain the field work 24 hours a day [89].

6.2 Queueing Theory to Analyse the Proposed System Outputs

6.2.1 Queueing Theory Introduction

Queueing theory is a method to mathematically describe the behaviour of waiting lines or queues [90]. A queue is generally formed by people or things such as phone calls, robots, requests, or materials which are waiting to be served. A simple queuing system has at least three elements which are: an arrival rate, a service rate, and servers. Queueing theory is an applied probability field to determine the arrival rate and service waiting time. A.K. Erlang published the first paper in queuing theory in 1909 when he studied the effect of the number of available telephone circuits and customers' waiting time to make calls [91]. The queueing analysis helps in estimating the optimal serving time, the required number of servers, and servers utilization [92].

Kendall's notation is used to represent any queueing system. It denotes the inter-arrival distribution function, service time distribution function, number of available servers, the system's capacity, population size, and service discipline. The notation *M /M /m* is the most common notation used for simple queueing systems. The inter-arrival *M* is assumed to be independent and follows a Poisson distribution, the service waiting time *M* is also independent follows an exponential distribution, and there is *m* number of serving RTAs. The simple queueing system is assumed to have an infinite capacity, infinite population, and a service discipline. The notations used for the queue analysis are as follows:

- λ is the dispatching requests arrival rate per unit time

- α is the requests service rate
- ρ is the RTAs utilization
- m is the number of RTAs
- L is the mean number of requests in the system which are waiting to be served and the one that are being served
- L_q is the mean number of requests in the queue waiting to be served
- W is the mean of the waiting time in the system which is the time the request spends in the queue waiting to be assigned to a an RTA and the time the request waits for the RTA to arrive
- W_q is the mean of the waiting time in the queue to be assigned to be served by an RTA
- $P(0)$ is the probability of 0 requests in the system
- $P(x)$ is the probability of x requests in the system
- C_w is the cost of a service waiting time
- C_s is the operational cost of RTAs per unit time.

The mathematical formulation used to analyse a single-server *M /M /1* queue [93] are as follows:

$$\rho = \frac{\lambda}{\alpha} \tag{6.1}$$

$$L_q = \frac{\rho^2}{1-\rho} \tag{6.2}$$

$$L = L_q + \frac{\lambda}{\alpha} \tag{6.3}$$

$$W_q = \frac{L_q}{\lambda} = \frac{\rho^2}{\lambda\,(1-\rho)} \tag{6.4}$$

$$W = Wq + \frac{1}{\alpha} \tag{6.5}$$

$$P(0) = 1 - \frac{\lambda}{\alpha} \tag{6.6}$$

$$P(x) = \left(\frac{\lambda}{\alpha}\right)^{x} P(0) \tag{6.7}$$

The mathematical formulation used to analyse a multiple-server *M /M /m* queue [93] are as follows:

$$P(x) = \begin{cases} \left[\left(\sum_{x=0}^{m-1} \frac{\rho^x}{x!}\right) + \frac{\rho^m}{(m)!\,(1-\rho)}\right]^{-1} & for\ x = 0 \\ \frac{\rho^x}{x!}\, P(0) & for\ 1 \le x \le m \\ \frac{\rho^x}{m!\, m^{x-m}}\, P(0) & for\ x > m \end{cases} \tag{6.8}$$

$$Lq = \left[\frac{\rho^{m+1}}{(m-1)!\,(m-\rho)^2}\right] P(0) \tag{6.9}$$

$$L = \rho + Lq \tag{6.10}$$

$$Wq = \frac{Lq}{\lambda} \tag{6.11}$$

$$W = \frac{L}{\lambda} \tag{6.12}$$

$$\text{Total Cost} = C_w \text{L} + C_s \text{m} \quad (6.13)$$

6.2.2 Applying the Queueing Theory to Analyse the Harvest and Yield Management Automation System

The harvest and yield management automation system is a queueing system which has HPAs who send dispatching requests, wait to be served by RTAs (service waiting time), and finally release the RTAs. Thus, the system represents a simple queueing system. The main objective of analysing the system's queueing property is to deliver an immediate or minimum service waiting time. Arbitrarily increasing the number of serving RTAs could improve the service waiting time, but it is not economical and may lead to accumulated and under-utilized RTAs. On the other hand, decreasing the number of severing RTAs leads to long service waiting time. The queueing theory balances the minimum number of the required RTAs and service waiting time.

The system queueing analysis requires understanding the arrival of the dispatching requests and service waiting time probabilistic properties which were presented in Section 5.5.1. The RTAs considered to be serving the HPAs concurrently meaning that the system has two queues with multiple RTAs. This queueing system model Kendall's notation is *M /M /m* and it has an infinite capacity, infinite population, and FIFO service discipline.

This system has two queues which are the request queue and the available RTAs queue. The request queue holds arriving requests while the available RTAs queue holds the RTAs. The request queue service discipline is FIFO to consider serving the first arriving request, but the available RTAs queue discipline is dynamic which will be investigated in this chapter. The objective of studying the available RTAs queue and how the central controller selects the RTAs is to reduce the service waiting time and fully utilise the RTAs.

The dispatching requests get added to a queue and wait for service. The way the system algorithm assign the RTAs to handle queueing requests and how these requests behave while waiting for service determine the queue characteristics. The selection of the RTAs from the available RTAs queue is dynamic. The central controller handles the dispatching requests and the RTAs' selection as shown in Fig. 6.3.

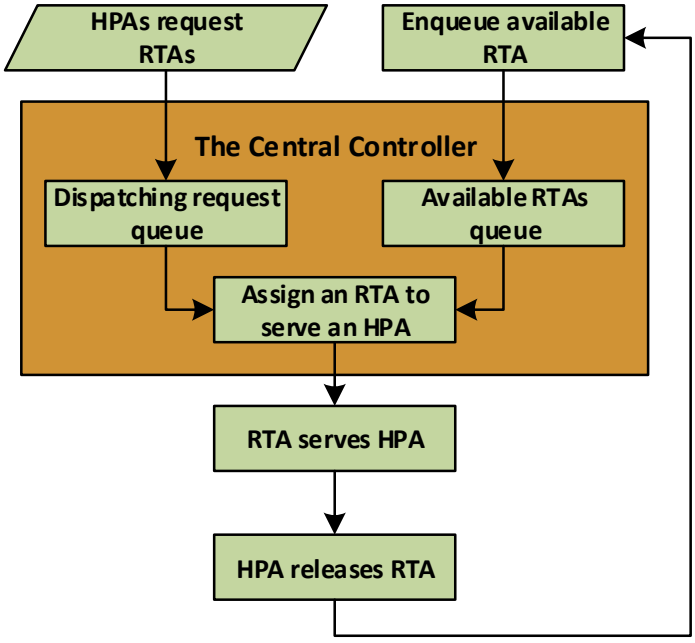


Fig. 6.3. The automated yield management system mechanism.

6.2.3 The Proposed System Queueing Analysis Setup

The observed service waiting time in this system is the duration between when an HPA makes a dispatching request until the RTAs arrives at the HPA's location to serve that request. The time which the RTAs spends while serving the HPAs is neglected since it is the same time spent by the HPAs to empty their picking bags to fixed bins.

This queueing system can be considered as a single-server *M /M /1* queue since the average of service waiting time was observed regardless of the number of deployed RTAs, and thus the whole system is considered as a single server. However, the system can also be considered as

a multiple-server *M /M /m* queue system since each RTA represents a single server. The multiple-server queue is assumed to have identical servers, and the mean of the service waiting rate will be observed for all servers.

6.3 Using Queueing Theory to Analyse the Single RTA Model and the Two Small RTA models Simulation Scenarios using the FAFS Dispatching Algorithm

The single RTA model simulation scenario presented in Section 5.5 is single-server *M /M /1* queueing model with the mean of the dispatching requests arrival rate $\lambda = 0.92$ per min and its constant average service rate $\alpha = 20$ per min. The RTA utilization factor ρ = 4.6%, the number of queuing requests L_q = 0.0022, and the number of requests in the system L = 0.0482. The mean service waiting time in the queue $W_q = 0.144s$ and service waiting time in the system $W = 3s$ which is similar to the mean of the service waiting time. The probability the RTA being idle, or there were no dispatching requests in the queue $P(0) = 95.4\%$ and the likelihood there were three requests was $P(3) = 0.0093\%$. The queueing analysis results showed similar results to the service time reliability analysis. However, it showed that RTAs was not fully utilized which means that the system could serve more HPAs and still have a similar service waiting time.

The second simulation scenario which deployed two small RTA presented in Section 5.6 is considered as *M/ M/ m* queueing model with $\lambda = 0.92$ per min and $\alpha = 15.8$ per min. The model utilization factor $\rho = 5.82\%$. The possibility that an RTA was idle or there were not dispatching requests in the queue $P(0) = 94.3\%$ while $P(3) = 0.08\%$. The number of the queuing requests $L_q = 4.93 * 10^{-5}$ and the number of requests in the system $L = 0.06$. The service waiting time in the queue $W_q = 0.0032s$ and the service waiting time in the system $W = 3.9s$.

When comparing the queueing analysis results of the single RTA and two-RTA scenarios, the deployment of two RTAs improved the RTAs utilisation factor from 4.6% to 5.8% meaning that the higher the utilisation the more efficient the RTAs. The two-RTA scenario improved the service waiting time from 0.144s to 0.0032s.

The queueing analysis provided more details about the system behaviour compared to the RTAs service time reliability analysis. For example, when deploying multiple small RTAs, the requests spend less time in the request queue. Moreover, it confirmed the RTAs service time reliability analysis results since the longer service waiting time was caused by the mechanism the central controller used to selects available RTAs since the requests spent less time in the queue compared to the single RTA scenario.

6.4 First Available First Serves (FAFS) Dispatching Algorithm

6.4.1 FAFS Dispatching Algorithm Work Concept

The FAFS algorithm work concept is based on the First in First out (FIFO) queueing discipline which was used in Chapter 5. The RTAs are shared resources which the HPAs queue up to use. In the simulation, The HPAs acquire RTAs services by sending dispatching requests. The RTAs become available or get released after serving the HPAs.

The FAFS algorithm initializes two queues, which are a request queue and RTAs availability queue. The RTAs availability queue holds the available RTAs' IDs. The maximum number of elements in the RTAs availability queue corresponds to the total number of RTAs deployed in the system and initially located at the drop station. The request queue holds incoming dispatching requests. When the central controller receives a dispatching request, it checks if the RTAs availability queue is not empty. The central controller dequeues the first available

RTA’s ID and assigns the RTA which corresponds to the dequeued RTA’s ID to serve the first request which is dequeued from the request queue.

The RTAs enqueue their IDs back to the availability queue after serving the HPAs. If the RTAs availability queue is empty, the algorithm will wait for an RTA to be released. The service waiting time starts when an HPA makes a dispatching request and ends when an RTA arrives at the HPA’s location to dispatch the yield. The simulation sequence diagram shown in Fig. 6.4 illustrates the FAFS dispatching algorithm concept.

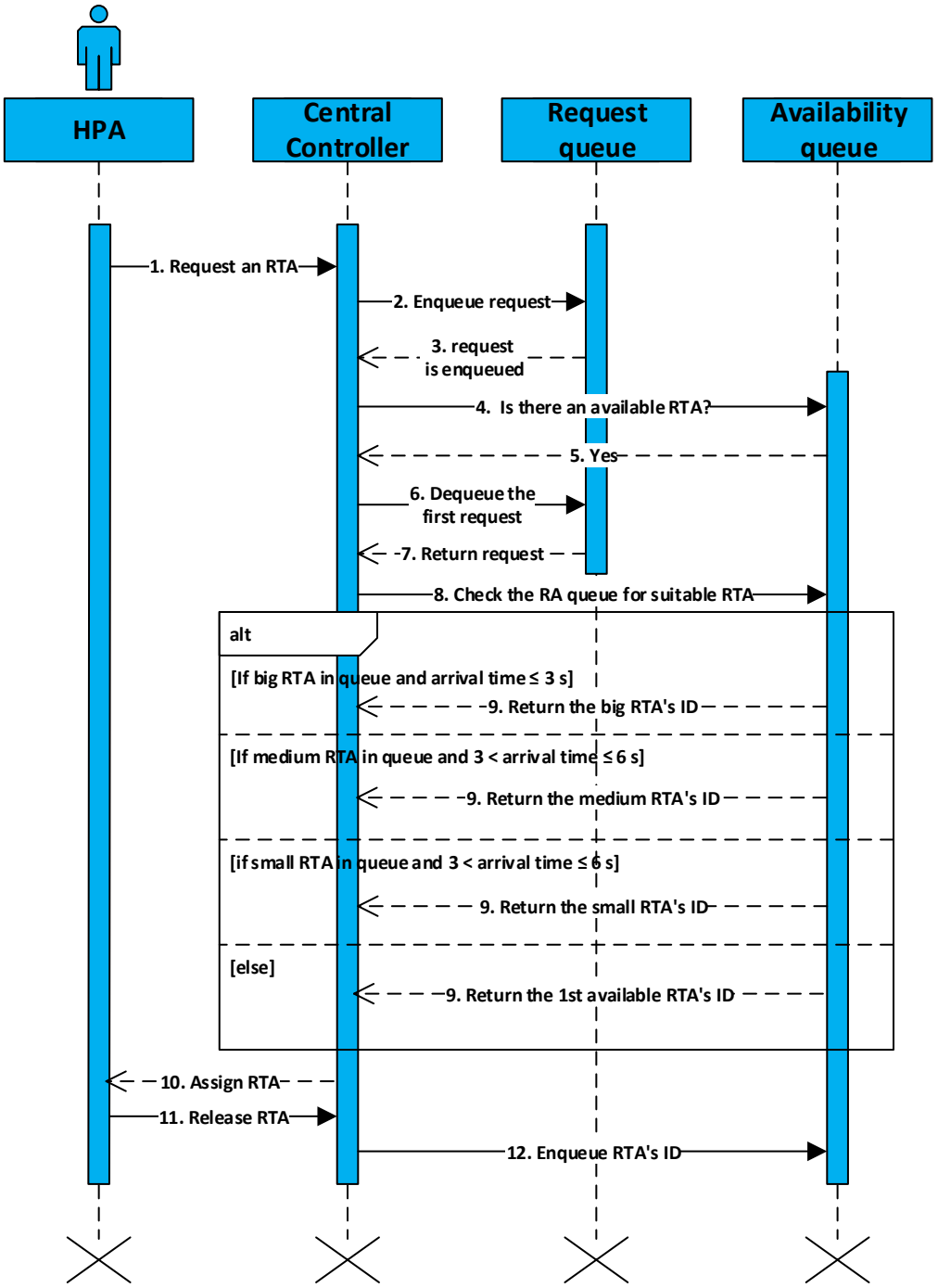


Fig. 6.4. The FAFS algorithm's sequence diagram.

6.4.2 FAFS Dispatching Algorithm Simulation Setup

The queueing analysis for the automated yield management system discussed above showed that the system was not fully utilized when a single RTA served three HPAs. When the system deployed two small RTAs, they were barely utilized. Thus, this simulation scenario deployed

three small RTA models to serve ten HPA models, since in reality eight pickers are required per block. As a result, we hypothesised that there should be at least one RTA to serve three HPAs and to add more variation to the simulation. Each RTA model had a payload capacity of 200 kg a speed of 2 m/s. The HPA models attributes are shown in Table 6.1. The block model was similar to the block model simulated in Section 5.2.2.

The Monte Carlo method was used in this scenario to repeatedly simulate the scenario 10000 times similar to the pervious scenarios. The system service time reliability was also analysed by fitting the data to the Kaplan-Meier estimator. Moreover, the queueing analysis was used to assess the FAFS algorithm performance and the RTAs efficiency.

6.4.3 FAFS Dispatching Algorithm Simulation Results

The HPA models results summary are listed in Table 6.1 while their picking bags filling time and the service waiting time distribution are shown in Appendix C1. The HPA models who had better experience were faster, and visited more trees. There was not any correlation between the picking bag's weight threshold value and the HPA models performance. However, the threshold value had a correlation with the number of dispatching requests and the number of RTA models dispatching trips. The smaller the threshold value, the more frequent the system receives dispatching requests. It is more efficient to set the picking bag weight threshold to the maximum recommended value.

Table 6.1. The HPAs attributes and simulation results including yield quantity, bags filling time, and service waiting time.

HPA's ID	Picking skills level	Picking bag's weight threshold value	No. visited trees	No. filled bags	Max. bag's filing time in s	Min. bag's filing time in s	Mean bag's filing time in s	Max. service waiting time in s	Min. service waiting time in s	Mean service waiting time in s
HPA1	Beginner	25 kg	24	349	358.6	330.9	344.7	54.3	1	10.1
HPA2	Beginner	20 kg	24	437	289.1	262	275.7	49.4	1	9.97
HPA3	Beginner	15 kg	24	580	217.6	191.2	207.1	49.6	0	9.4
HPA4	Moderately skilled	25 kg	34	493	251.6	229.8	240.2	49.3	1	8.6
HPA5	Moderately skilled	20 kg	34	622	201.2	183.3	192.1	51.7	1	8.5
HPA6	Moderately skilled	15 kg	34	812	154.5	136	144.4	50.2	1	8.3
HPA7	Highly skilled	25 kg	45	649	191.3	175.5	182.7	50.1	1	8.8
HPA8	Highly skilled	20 kg	45	805	153.2	138.9	146.3	48.9	1	8.8
HPA9	Highly skilled	20 kg	44	789	153.5	137.8	146.2	46	1	8.4
HPA10	Highly skilled	15 kg	44	1055	115.7	103.5	109.8	45.9	0	8.5

The HPA models made 6600 dispatching requests. The inter-arrival of dispatching requests was 0-253.2s with the mean of 19.01s as shown in Appendix C1. The distribution of the dispatching requests inter-arrival was smoother than the previous simulations distribution due to the increase in the dispatching requests made by the HPA models causing the period between requests arrival relatively smaller. It was observed that the increase in the weight threshold value caused the service waiting time to increase due to the fact that the HPA models who had smaller threshold value made more dispatching requests. As a result, the RTA models were at a close proximity to the HPA models with smaller threshold value. The dispatching requests arrival rate $\lambda = 3.12$ per min which follows a Poisson distribution as shown in Fig. 6.5.

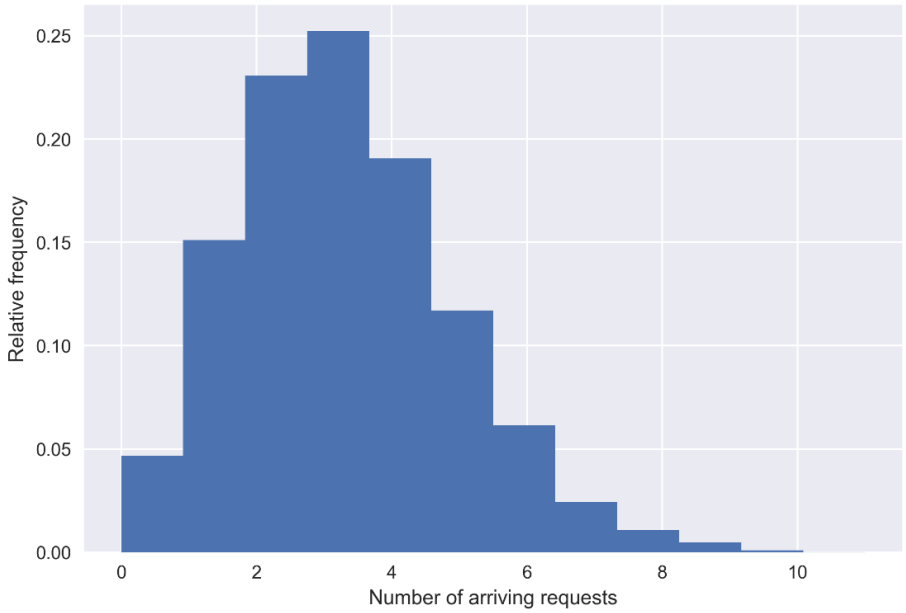


Fig. 6.5. The distribution of the dispatching requests arrival.

The RTA models dispatched 6600 picking bags. The HPA models service waiting time was 0-54.3s with the mean of 8.8s. The constant average service rate $\alpha = 6.83$ per min. The system had the capacity to handle nearly seven requests per min. The process time of each request per a unit time was continuous and independent, so service rate was assumed to follow an exponential distribution. The generated service rate distribution is shown in Fig. 6.6.

The Monte Carlo method estimated the mean of the inter arrival of dispatching requests as 19.1s and the mean of service waiting time as 8.76s. Thus, the mean arrival rate of dispatching requests $\lambda = 3.14$ and the constant average service rate $\alpha = 6.85$.

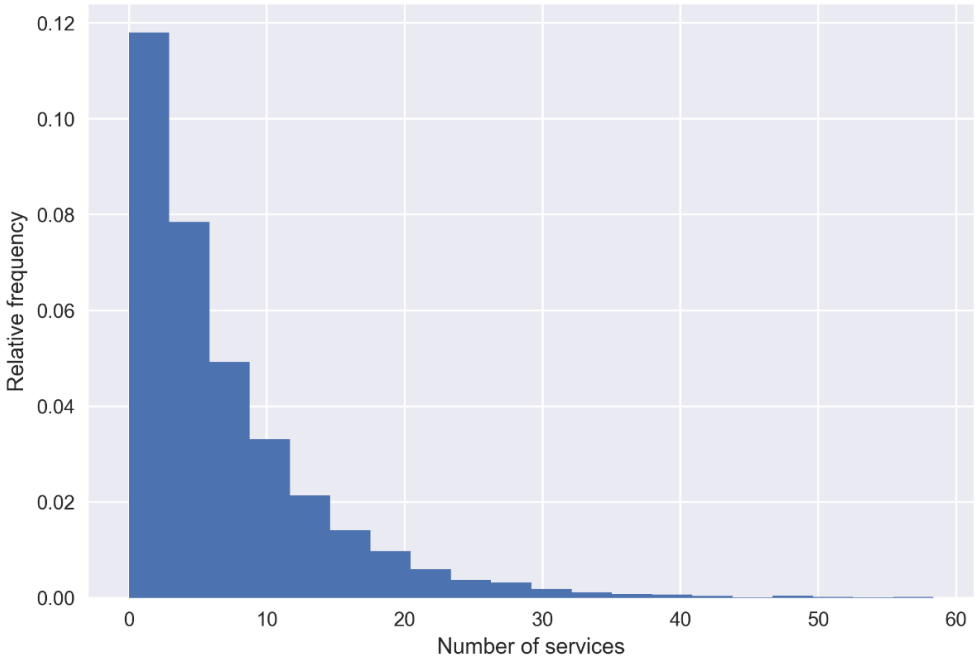


Fig. 6.6. Service rate distribution per minute.

The Kaplan-Meier estimator plot estimated unserved requests at 0s as $\hat{R}(0) = 0.996$ since only 25 requests out of 6600 were dispatched instantly. Thus, there was a 0.4% chance that the system could immediately serve a request. Moreover, the $\hat{R}(1) = 0.85$ indicating that there was a 15% chance to handle a request within 1s. The Kaplan-Meier plot has a steep decline from 0 to 10s, and a smooth decline afterward. The system is considered acceptable since there was only a 20% chance that service an HPA model would wait for 11s. The successful requests dispatching function estimator plot for each HPA is shown in Appendix C2.

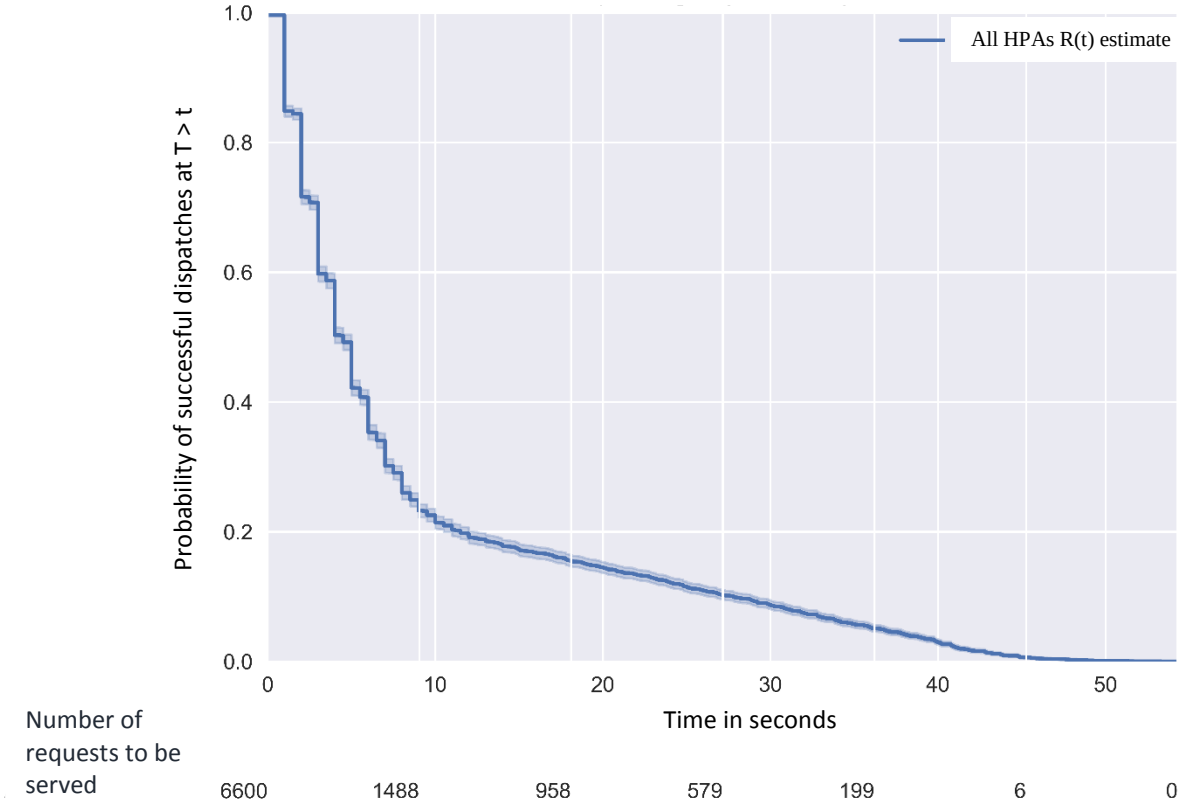


Fig. 6.7. Kaplan-Meier plot estimator for the FAFS scenario with 3 RTA models serving 10 HPA models. The first vertical line on the grid represents the number of unserved dispatching requests shown under the time axis.

There are noticeable differences between the simulation scenario employing two 400 kg RTA models in Section 5.4 and this simulation scenario. In this scenario, service waiting time was long with a mean of 8.8s, maximum waiting time = 54.3s, and only 25 requests were handled instantly. However, the RTA models were 50% lighter which reduced the soil compaction at the same time managed to serve 10 HPA models.

This simulation scenario had an *M/ M/ m* queueing system and the RTAs utilization factor $\rho = 0.46$. The probability that there were not dispatching requests in the queue $P(0) = 63\%$. The likelihood that the three RTA models are busy and there are three dispatching requests on the queue $P(3) = 1\%$. The likelihood that there would ten dispatching requests, which is the maximum number of requests that can be made in this scenario, waiting to be served and that all the RTA models were busy $P(10) = 1.91 \times 10^{-6}\%$. The number of queuing requests in

the queue $L_q = 0.0021$, and the number of requests in the system $L = 0.46$. The service waiting time in the queue $W_q = 0.042s$, and service waiting time in the system $W = 9s$.

The queueing waiting time was less than 1s, and the RTAs utilization factor had improved even though the three RTA models were smaller and served more HPA models. The request waiting time in the request queue was significantly low. The queueing analysis showed that the three small RTAs could efficiently serve up to 21 HPAs and still maintain a relatively short service waiting time.

The total harvest time in this simulation scenario was 34.89 h, 4.4 working days, which was 70% less than the simulation results shown in Section 5.7. The total service waiting time for all HPAs was 16.1h which exhibits a significant increase compared to Section 5.7. However, service waiting time made 46% of the total harvest time. The increase in the service waiting time is due to the FAFS dispatching algorithm work concept which assigns RTAs based on their availability only and neglects their locations.

The service waiting time varies and it could take longer than expected. The main reason for having a longer service waiting time is due to the RTAs travelling time when navigating from their location to the HPAs. Even though, the system employed the Dijkstra's algorithm to determine the shortest path.

The FAFS algorithm assigns the first available RTA regardless of their location. However, the FAFS algorithm analysis showed that the system manages assigning the RTA models efficiently and that the RTAs availability was not the issue. As a consequence, the long service waiting time was caused by the Moreover, the queueing analysis showed that the system was not fully utilized even when serving ten HPA models.

6.5 Dynamic Distance (DD) Dispatching Algorithm for Homogenous RTAs

6.5.1 DD Despatching Algorithm Work Concept

The DD despatching algorithm helps the central control to assign the available homogenous RTAs based on their location, preferred the closest available RTA to the incoming dispatching request. This selection method shortens the service waiting time.

The algorithm initializes two queues, which are a request queue holding incoming dispatching requests, and an RTAs availability queue holding available RTAs' IDs. As explained in the RTA model Section 4.3.4, the RTAs' IDs are the keys to a dictionary that holds and keeps track of the RTA's attributes including their locations. The RTAs are initially located at the drop station. The dispatching requests are added to the request queue and if the RTAs availability queue is not empty, it will check how many RTAs are in the queue. If there is only one RTA's ID, the algorithm dequeues the RTA's ID, dequeues the first dispatching request, and assigns the RTA corresponding to the RTA's ID to handle the request. If the availability queue has more than a single RTA, the algorithms will iterate the queue, calculating the distance from each RTA's position to the request's position, and selects the closest RTA to the request location.

The RTAs enqueue their IDs back to the RTAs availability queue after serving the HPAs. If all RTAs are busy, the algorithm waits for any HPA to release an RTA. The simulation sequence diagram in Fig. 6.8 illustrates the DD dispatching algorithm work concept.

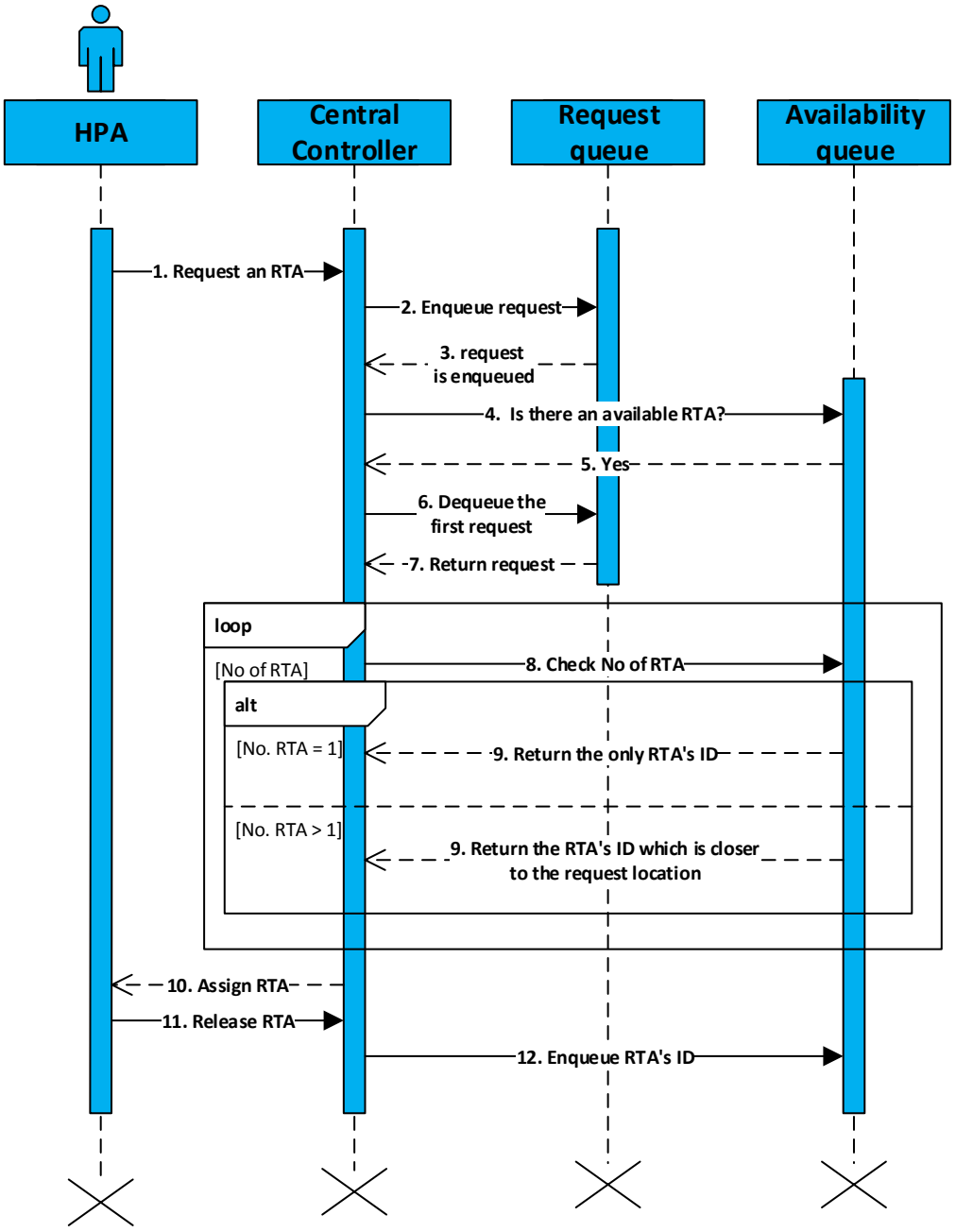


Fig. 6.8. The DD dispatching algorithm sequence diagram.

6.5.2 DD Despatching Algorithm Simulation Setup

This simulation scenario had the same setup as the simulation in Section 6.4 for easy comparison. It had the same HPA models, the same RTA models and the orchard block model. The same methods were used to analyse results, including service time reliability analysis and queueing analysis. However, the DD algorithm results were statistically compared to the FAFS

algorithm simulation results. The long-rank test was applied with a statistical significant p-value < 0.01. The log-rank test was chosen since the resulted service's waiting time were highly skewed through the whole observation period. The log-rank test had a confidence value α = 0.99, a single degree of freedom, and the Chi-squared null distribution.

6.5.3 DD Despatching Algorithm Simulation Results

The HPA models attributes and simulation results are listed in Table 6.2. They made 6603 dispatching requests and their picking bags filling time and service waiting time results are plotted and shown in Appendix C3. There are 3 more bags compared with the FAFS algorithm results due to the response of the system while employing different algorithm causing the random pseudo generated number sequence to be distributed differently among the HPAs. The time difference between the arrivals of two consecutive dispatching requests, shown in Appendix C3, took 0-344.7s with a mean of 18.4s. The request constant average arrival rate λ was 3.26 per min. The requests arrival was independent and was assumed to fit a Poisson's distribution as shown in Appendix C3.

Table 6.2. The HPAs attributes and simulation results including yield quantity, bags filling time, and service waiting time.

HPA's ID	Picking skills level	Picking bag's weight threshold value	No. visited trees	No. filled bags	Max. bag's filing time in s	Min. bag's filing time in s	Mean bag's filing time in s	Max. service waiting time in s	Min. service waiting time in s	Mean service waiting time in s
HPA1	Beginner	25 kg	24	350	344.6	332.3	344.9	32.2	0	3.3
HPA2	Beginner	20 kg	24	431	289.1	263.1	275.7	36.9	0	3.3
HPA3	Beginner	15 kg	24	575	219.2	194.1	206.9	14.5	0	2.9
HPA4	Moderately skilled	25 kg	34	497	252.7	231.3	240.2	33	1	2.9
HPA5	Moderately skilled	20 kg	34	622	201.4	183	192.2	38.6	0	2.8
HPA6	Moderately skilled	15 kg	34	814	154.8	135.8	144.3	41.2	0	2.8
HPA7	Highly skilled	25 kg	45	652	191.6	175.5	182.7	33.9	0	2.6
HPA8	Highly skilled	20 kg	44	801	153.2	138.2	146.3	42.9	0	2.8
HPA9	Highly skilled	20 kg	45	802	154.4	139.7	146.2	25.8	0	2.5
HPA10	Highly skilled	15 kg	44	1059	115.9	102.3	109.8	40.8	0	2.5

The service waiting time took 0-42.9s with the mean of 2.8s. The constant average service rate α was 21.4 per min and service time was independent and continuous which was assumed to have an exponential distribution as shown in Appendix C3. The DD dispatching algorithm improved the service waiting time mean and constant average service rate by 68.1% compared to the FAFS dispatching algorithm. The Monte Carlo method estimated the mean of the time between the arrivals of two dispatching requests as 18.46s for and the service waiting time mean as 2.78s.

The DD dispatching algorithm reliability was estimated by the Kaplan-Meier estimator which is represented by the blue curve as shown in Fig. 6.9. The figure also displays the FAFS dispatching algorithm reliability analysis, the green curve, for comparison. The Kaplan-Meier curve had a steep decline from 0-5s. The estimated unserved requests $\hat{R}(0) = 0.98$ since only 128 out of 6603 requests were immediately served, or the chance to serve a request at 0s was a 2%. The $\hat{R}(1) = 0.65$ indicating that there was a 35% chance to dispatch a bag within 1s. The DD dispatching algorithm system is more reliable compared to the FAFS algorithm since there was only a 6% chance that the service waiting time might take more than 6s. The reliability function estimator plot for each HPA model is shown in Appendix C4.

Fig. 6.9 shows a clear difference between the DD and FAFS dispatching algorithm. The log-rank test results indicates that the performance was statistically significantly different ($p < 0.01$, test statistic = 2715.6).

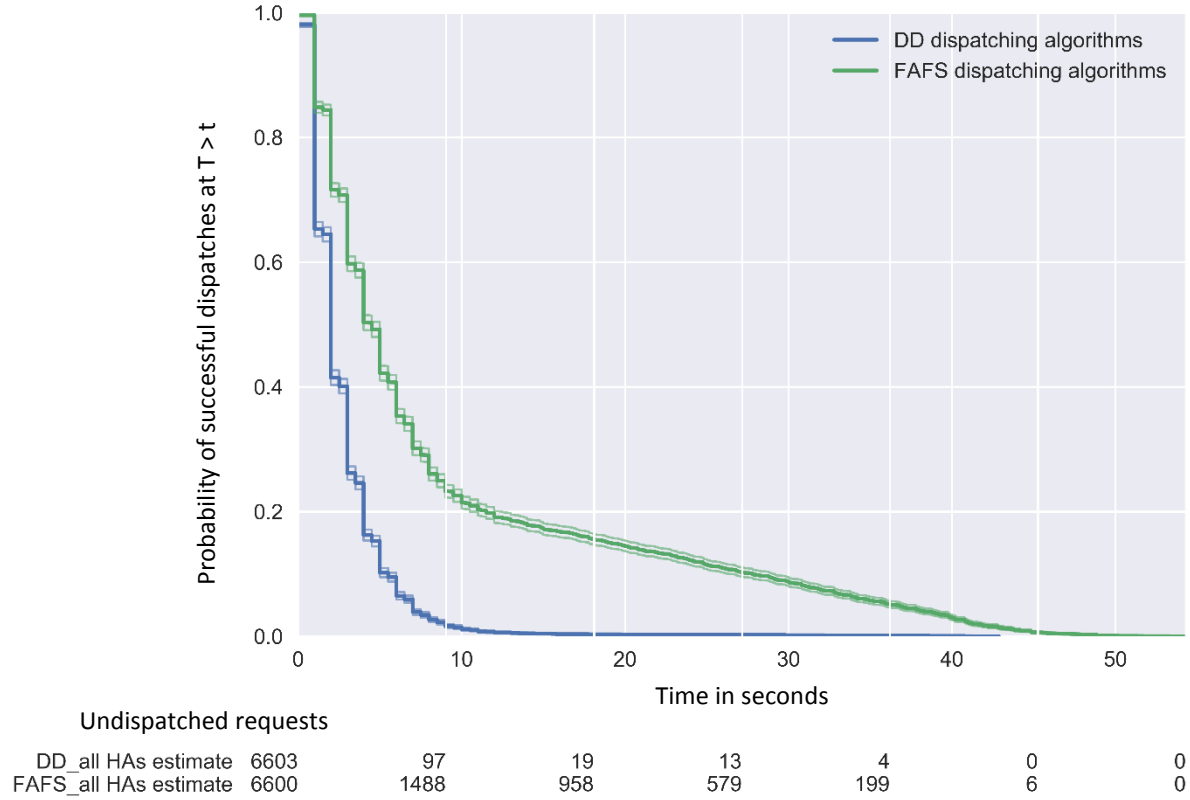


Fig. 6.9. The DD dispatching algorithm (blue curve) compared with FAFS dispatching algorithm (green curve) reliability analysis estimate.

The DD dispatching algorithm queueing system is an *M/ M/ 3*, and the RTAs utilization factor $\rho = 0.15$ calculated by Eq. (6.1). The probability that there were not dispatching requests in the queue $P(0) = 61\%$, three dispatching requests $P(3) = 0.034\%$, ten dispatching requests $P(10) = 1.6 \times 10^{-5}$ %. The number of queueing requests in the queue L_q= 0.0002, and the number of requests in the system L was 0.15. The waiting time in the queue was W_q= 3.5×10^{-4}s, and the service waiting time in the system $W = 2.76$s. The queueing waiting time was less than 1s, and the RTAs utilization factor had significantly improved.

The RTAs utilisation factor $\rho = 0.15$ which means that it still has the capacity to serve more HPA models without causing long waiting queue. The system maximum capacity is investigated in Section 6.5.5.

The total harvest time was 33.85h. However, the harvest can be handled in 24h by increasing the number of HPAs. The total service waiting time for all HPAs was 5.1h which improved the total service waiting time by 68.3% when compared to the FAFS algorithm. However, service waiting time made 15% of the total harvest time.

6.5.4 The Monte Carlo Convergence Analysis

6.5.4.1 Convergence Analysis Setup

The Mote Carlo convergence analysis is conducted to determine the minimum number of required simulations to achieve an acceptable level of accuracy. The same scenario in Section 6.5.1 was simulated 7 times as follows:

1. A single run of Monte Carlo simulation
2. 10 runs of Monte Carlo simulation
3. 50 runs of Monte Carlo simulation
4. 100 runs of Monte Carlo simulation
5. 500 runs of Monte Carlo simulation
6. 1000 runs of Monte Carlo simulation
7. 10000 runs of Monte Carlo simulation

The service waiting time was observed and the algorithm took the mean value the service waiting time of each simulation run. The resulted values were plotted as a boxplots to determine how the average values of each Monte Carlo simulation were spread out. All the 7 simulations are plotted together for comparison.

6.5.4.2 Convergence Analysis Results

The Monte Carlo assessment test results summery are presented in Table 6.3 while Fig 6.10 shows and compares the simulations data as they were plotted in box and whisker plots.

Table 6.3. The summery of the Monte Carlo convergence analysis for service waiting time.

	1_sim	10_sim	50_sim	100_sim	500_sim	1000_sim	10000_sim
mean	2.848	2.770	2.778	2.783	2.786	2.789	2.786
std	0.286	0.028	0.045	0.045	0.047	0.049	0.050
min	2.459	2.726	2.681	2.681	2.665	2.661	2.631
25%	2.689	2.748	2.745	2.749	2.754	2.755	2.751
50%	2.819	2.777	2.776	2.780	2.783	2.787	2.785
75%	2.919	2.792	2.806	2.810	2.817	2.822	2.820
max	3.327	2.806	2.892	2.899	2.945	2.945	2.997

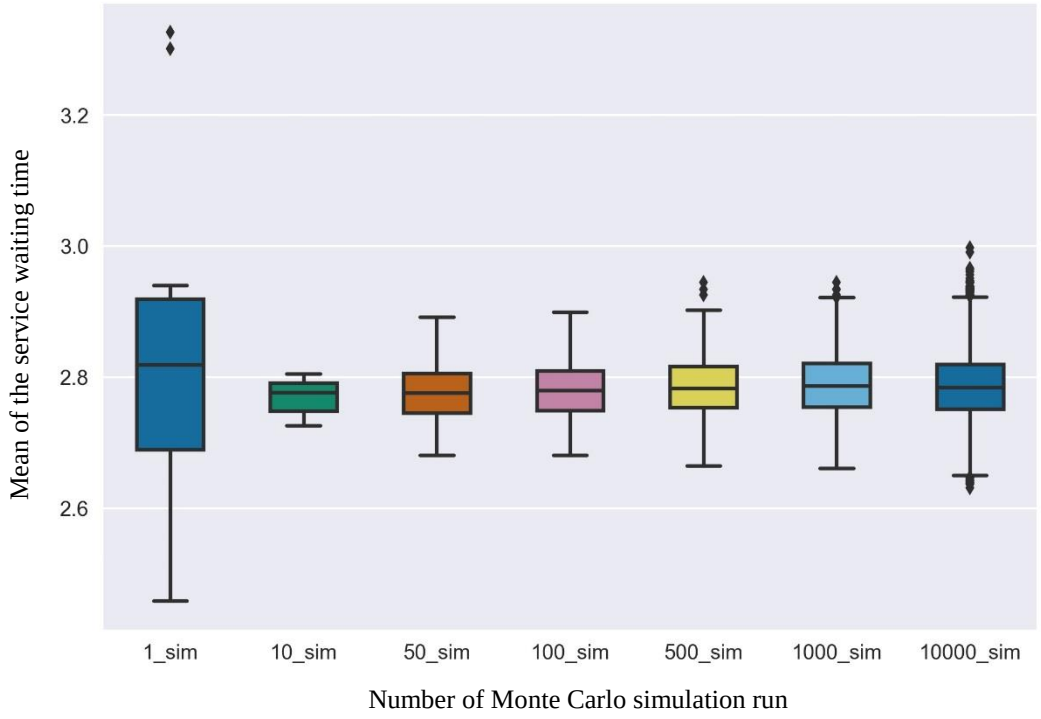


Fig. 6.10. Boxplot showing the Monte Carlo simulation convergence analysis.

The results shows that the first simulation which had a single run is unsymmetrical, spread out, and has a number of outliers. The second simulation which had 10 runs appears to be converging since it has small spread, but it is still somewhat unsymmetrical. The third simulation which had 50 runs has a symmetrical spread without outliers. The remaining simulations have similar results to the third simulation. From the table, the spread of 5-1000

runs is approximately the same. Thus, 50 is the threshold for the number of Monte Carlo simulation runs whenever results needs further analysis such as the system capacity and cost optimization. However, in some Monte Carlo simulation scenario 10000 runs were used sometimes since the large number of simulations provide better approximation.

6.5.5 Analysis of the System Service Capacity while Using the DD Algorithm

This section analyses the system capacity while using the DD algorithm by simulating three 200kg RTA models with a speed of 2 m/s. This analysis conducted four simulations and on each simulation, the number of HPA models was increased as shown in Table 6.4 to determine many HPAs could be serviced by three RTAs and if the scaling was linear. These simulations were Monte Carlo simulation with 50 runs. The simulation results are presented in Table 6.4. Fig. 6.11 shows the relationship between the HPA models and the RTAs utilisation factor.

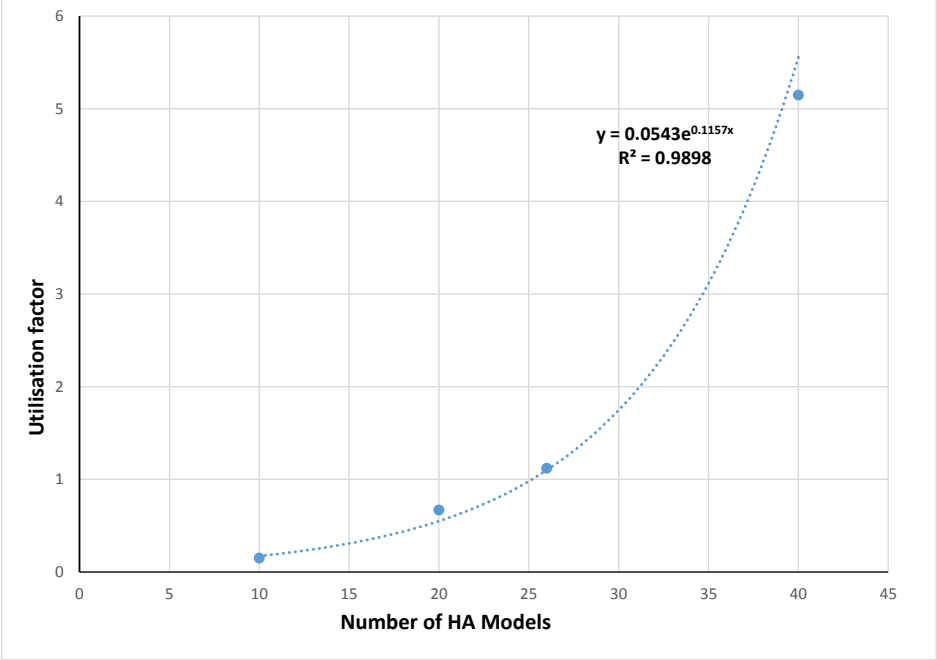


Fig. 6.11. The exponential relationship between the utilisation factor and the number of HPA models.

Table 6.4. The system capacity analysis results while simulation the DD algorithm with 3 RTA models.

No. of HPA	Experience	Threshold Weight in kg	Mean of waiting time in s	Mean of dispatching inter-arrival in s	λ per 1 min	α per 1 min	ρ
Test 1 with 10 HPA models							
1	Highly skilled	25	2.8	18.4	3.26	21.4	0.15
2	Highly skilled	20					
1	Highly skilled	15					
1	Moderately skilled	25					
1	Moderately skilled	20					
1	Moderately skilled	15					
1	Beginner	25					
1	Beginner	20					
1	Beginner	15					
Test 2 with 20 HPA models							
2	Highly skilled	25	6.4	9.5	6.3	9.4	0.67
4	Highly skilled	20					
2	Highly skilled	15					
2	Moderately skilled	25					
2	Moderately skilled	20					
2	Moderately skilled	15					
2	Beginner	25					
2	Beginner	20					
2	Beginner	15					
Test 3 with 26 HPA models							
4	Highly skilled	25	9.2	8.2	7.3	6.5	1.12
4	Highly skilled	20					
2	Highly skilled	15					
4	Moderately skilled	25					
2	Moderately skilled	20					
2	Moderately skilled	15					
4	Beginner	25					
2	Beginner	20					
2	Beginner	15					
Test 4 with 40 HPA models							
4	Highly skilled	25	29.4	5.8	10.3	2.0	5.15
8	Highly skilled	20					
4	Highly skilled	15					
4	Moderately skilled	25					
4	Moderately skilled	20					
4	Moderately skilled	15					
4	Beginner	25					
4	Beginner	20					
4	Beginner	15					
Test 5 with 60 HPA models							
6	Highly skilled	25	159.3	6.17	9.72	0.4	23.17
12	Highly skilled	20					
6	Highly skilled	15					
6	Moderately skilled	25					
6	Moderately skilled	20					
6	Moderately skilled	15					
6	Beginner	25					
6	Beginner	20					
6	Beginner	15					

The results show that the utilisation factor increases exponentially as the number of HPA models increases. The rule of thumb in queue theory that the $\rho < 1$ which means that

dispatching requests are arriving at a rate slower than they can be served. If the $\rho \geq 1$ then the queue line will grow without bound the same as the service waiting time would go to infinity.

Based on the results, this system has the capacity to serve up to 23 HPA models to maintain a maximum $\rho = 0.77$. Based on the queueing theory, the waiting time increases rapidly as the RTAs utilisation exceeds 80% [94]. Form the table, it can be seen that the service waiting time increased from 2-10s with increased HPAs and the dispatching requests came in faster 18.4-6.12s.

In reality, an orchard has several blocks and for every standard 4ha block there will be 8 HPAs to pick apple. The results show that three small RTAs can effectively serve up to 23 HPAs. Thus, three small RTAs are required for a 12ha orchard which has three blocks. The estimated number of RTA for every 4ha or for every 8 HPAs an RTA is required as shown in Fig. 6.12. However, the system capacity has a non-linear relationship to the number of RTAs due to their collaborative behaviour which increases the system capacity which was simulated and the results are presented in Table 6.5.

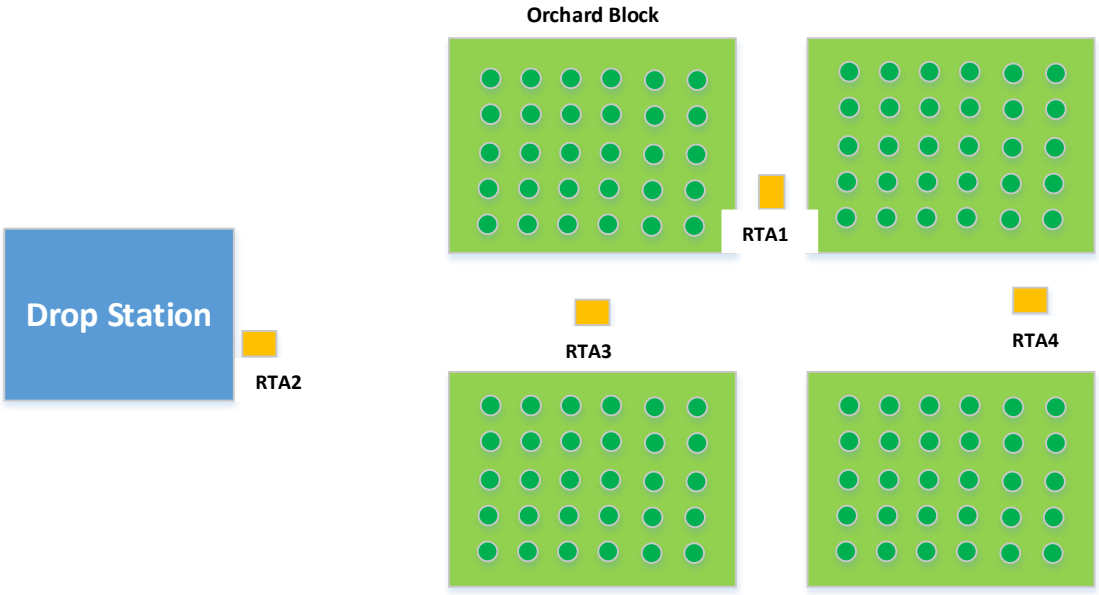


Fig. 6.12. 4 small RTAs are required for 4 blocks orchard with 32 HPAs picking apples.

Table 6.5. Simulation results for increasing number of RTAs and HPAs.

Number of HPAs	Number of RTAs	Mean of waiting time in s	Mean of dispatching inter-arrival in s	λ per 1 min	α per 1 min	ρ
10	3	2.8	18.4	3.26	21.4	0.15
20	3	6.4	9.5	6.3	9.4	0.67
30	4	9.3	6.6	9.1	6.4	1.4
40	5	11.3	5.04	11.9	5.3	2.2
50	6	13.3	4.1	14.6	4.5	3.2
60	7	14.5	3.5	17.1	4.1	4.2

The results in the table show that the increase of the number of HPAs is still causing an exponential increase to the utilisation factor, even though, the number of RTAs was linearly increased. The number of RTAs was linearly increase to maintain the RTAs utilisation less than 80%, but this did not help since the increase should have been exponential.

The DD dispatching algorithm successfully improved the service waiting time, number of RTAs, system utilization, and the soil damage. This algorithm is suitable for homogenous RTAs which have the same speed and payload capacity. The next section studies the effect of deploying heterogeneous RTAs with different speed and payload and how to select the most suitable RTA based on their speed and location.

6.6 Dynamic Distance and Best Fit (DDB) Dispatching Algorithm for Heterogeneous RTAs

The DDB despatching algorithm helps the central control to assign the available heterogeneous RTAs based on their location and their speed. The big RTAs are slow due to their size, limited manoeuvrability, and safety while the smaller RTAs are faster and have better manoeuvrability than big RTAs. It helps the central controller to assign the slow and big RTAs to serve nearby dispatching requests while assigning the small and fast RTAs to serve the requests at further distances. This selection method shortens the service waiting time by taking advantage of the big RTA payload capacity. Based on the previous simulation results, most of the dispatching requests are coming from a nearby distance to the RTAs.

The FAFS algorithm has shown that the deployment of smaller RTA models reduces the soil compaction. Furthermore, the DD dispatching algorithm has significantly improved the service waiting time and the RTAs utilization. However, the Single RTA model simulation analysis showed that deploying RTAs with larger payloads resulted in less service waiting time and less return trips to the drop station compared to deploying smaller RTAs. On the other hand, multiple small RTAs' collaborative behaviour enable them to compensate for the effect of frequent returns. This section investigate the design of dynamic dispatching algorithm which select heterogeneous RTAs based on their location and technical specification.

The algorithm assigns the best available RTA to handle a task at the shortest time possible. The algorithm will have the ability to evaluate each coming dispatching request at the same time assess the available RTAs with a low computation cost. This section explores the possibility to incorporate the advantages of big and slow RTAs with small and fast RTAs to further enhance the system reliability and optimize the service waiting time. Section 6.6.1 investigates each RTA efficiency and utilization.

6.6.1 DDB Despatching Algorithm Work Concept

The DDB dispatching algorithm selects the RTAs based on their location and speed. The algorithm starts by initializing a request queue to hold incoming request, and RTAs availability queue to hold available RTAs' IDs. The RTAs are initially located at the drop station, and they are not identical since they have different speed.

The algorithm dequeues the first dispatching request and checks its location then, checks each available RTAs locations and calculates the travel time to the request's location. The travel time is equal to the distance over the big RTA speed which is the slowest RTA. The slowest speed is chosen as a standard speed measure in order to consider the worst case scenario. The best available RTA is assigned to handle the request.

The algorithm selects the best RTAs by assessing each RTA in the RTAs availability queue. To select the best RTA, the algorithm checks the RTAs availability queue to satisfy one of the four following conditions:

1. If the big RTA (slow RTA) travel time is less than or equal to 3s, and the big and slow RTA is available then assign the big RTA
2. If the big RTA travel time is more than 3s and less than or equal to 6s, and the medium size and speed RTA is available then assign the medium RTA
3. If the big RTA travel time is greater than 6s, and the small and fast RTA is available then assign the small RTA
4. Assign the first available RTA in the queue if none of the three previous conditions were satisfied.

The RTA which satisfies one of the above conditions is assigned to serve the HPA. The dequeuing of a specific RTA's ID still follow the FIFO discipline by popping out the first RTA's ID in the queue and if the RTA does not meet the above condition, it will add them to the end of the queue. In other words, if another RTA's ID happens to appear before the required RTA in the RTAs availability queue, the algorithm will push it to the back of the queue. The algorithm use conditions and dequeuing methods instead of looping the queue since this requires less computation and has a linear growth. The RTAs re-queue their IDs after serving the HPAs. The simulation sequence diagram in Fig. 6.13 illustrates the DDB dispatching algorithm work concept.

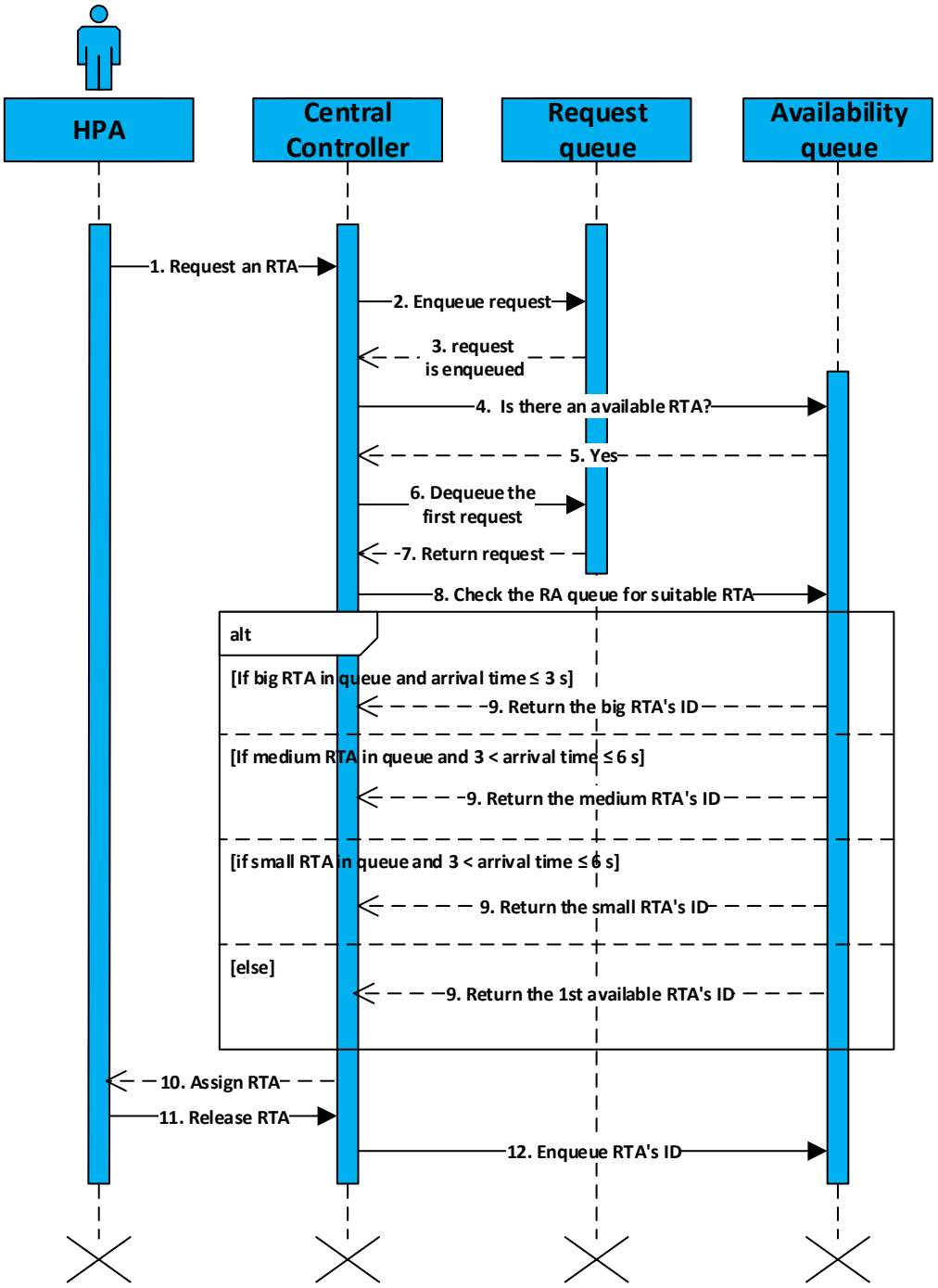


Fig. 6.13. The DDB dispatching algorithm sequence diagram.

6.6.2 DDB Despatching Algorithm Simulation Setup

The same number of HPA models and RTA models were simulated. The HPA models had the same attributes to the FAFS and DD dispatching algorithm simulation setup to compare their performances. However, the three RTA models in this simulation were heterogeneous and had

different payload capacity and speed. They were a big RTA model with a payload capacity of 800 kg and speed of 2 m/s; a medium RTA with a payload capacity of 400 kg and speed of 3 m/s; and a small RTA with a payload of 200 kg and speed of 4 m/s. The virtual orchard block was also similar to the one shown in Fig 5.1 which had 352 apple trees and a single drop station. The HPAs attributes and simulation results summary are listed in Table 6.2.

The same methods were used to analyse results, including service time reliability analysis and queueing analysis. The DDB algorithm results were statistically compared to the FAFS and the DD dispatching algorithm simulation results. The long-rank test was applied with a statistical significant p-value < 0.01. The log-rank test had a confidence value $\alpha = 0.99$, a single degree of freedom, and the Chi-squared null distribution. Moreover, each RTA's performance, utilization, and reliability were investigated.

6.6.3 DDB Despatching Algorithm Simulation Results

The HPA made 6481 dispatching requests. The picking bags filling time, and HPAs results are plotted and shown in Appendix C5. The number of picking bags produced by the HPA models was slightly different from the previous simulation scenarios. However, there was a significant drop in the number of bags which HPA6 produced as shown in Table 6.6. The other HPAs production had a small increase. The algorithm improved the maximum and average service waiting time.

The time difference between occurrences of two consecutive dispatching requests was 0-278.1s with a mean of 19.9s as shown in Appendix C5. The requests constant average arrival rate $\lambda = 3$ per min which was fitted to a Poisson's distribution as shown in Appendix C5. The service waiting time took 0-21.9s with the mean of 2.3s. The constant average service rate α was 26.1 per min and the service waiting time was fitted to an exponential distribution as shown in Appendix C5.

Table 6.6. The HPAs attributes and simulation results including yield quantity, bags filling time, and service waiting time when employing DDB dispatching algorithm.

HPA's ID	Picking skills level	Picking bag's weight threshold value	No. of visited trees	No. of filled bags	Max. bag's filing time in s	Min. bag's filing time in s	Mean bag's filing time in s	Max. service waiting time in s	Min. service waiting time in s	Mean service waiting time in s
HPA1	Beginner	25 kg	25	364	361.4	328	344.9	14	0	2.5
HPA2	Beginner	20 kg	26	465	288.9	264.5	275.6	17	0	2.6
HPA3	Beginner	15 kg	25	602	224.2	196.3	206.8	21.9	0	2.5
HPA4	Moderately skilled	25 kg	36	524	250.2	228.2	240.2	20.8	0	2.3
HPA5	Moderately skilled	20 kg	36	652	200.8	182.4	192.1	12	0	2.3
HPA6	Moderately skilled	15 kg	14	335	151.9	135.7	144	10.5	0	2.3
HPA7	Highly skilled	25 kg	47	681	190.4	174.5	182.7	10.4	0	2.1
HPA8	Highly skilled	20 kg	48	862	155.2	139.2	146.3	6.8	0	2.1
HPA9	Highly skilled	20 kg	47	852	153	138.4	146.3	20.4	0	2.3
HPA10	Highly skilled	15 kg	47	1124	116.5	103	109.8	9.9	0	2.1

The DDB dispatching algorithm significantly improved the service waiting time mean and the maximum waiting time by 21.1% and 49% when compared with the DD algorithm and by 75% and 60% when compared with the FAFS algorithm.

The DDB dispatching algorithm service time reliability estimate is plotted in the blue curve shown in Fig. 6.11. The estimated unserved requests $\hat{R}(0) = 0.98$ since 134 out of 6481 requests were served instantly. This means that there was a 2% chance to handle a request instantly. The $\hat{R}(1) = 0.67$ indicating that there was a 33% chance to handle a request within 1s, but the $\hat{R}(5) = 0.05$ or there was a 95% chance a request would wait for 5 seconds to be served which explains the Kaplan-Meier estimation curve having a steep decline from 0 to 5s. By applying the DDB algorithm, there is only a 1.5% chance that serving a request might take more than 6s. The reliability function estimator plots for each HPA are shown in Appendix C3.

Fig. 6.14 shows a clear difference between the DDB algorithm (blue curve) and the DD algorithm (green curve) by plotting both Kaplan-Meier estimators. The log-rank test indicates

that the performance was statistically significantly different with test statistic = 275.8 and $p <$ 0.00001.

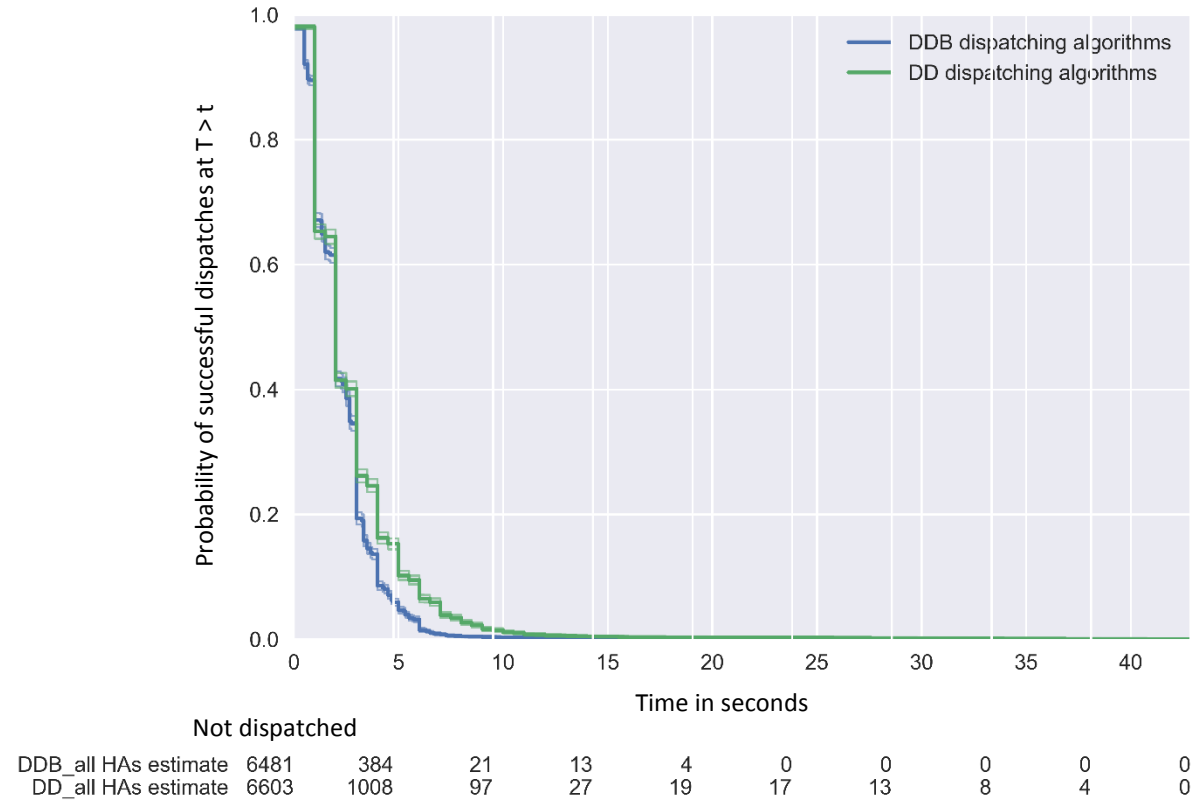


Fig. 6.14. DDB dispatching algorithm reliability analysis estimate (blue curve) compared with the DD dispatching algorithm estimate (green curve).

The DDB dispatching algorithm, the DD dispatching algorithm, and the FAFS dispatching algorithm performance was visually compared by plotting their Kaplan-Meier estimators together as shown in Fig. 6.15. The figure shows the improvement in the service waiting time and the system utilization.

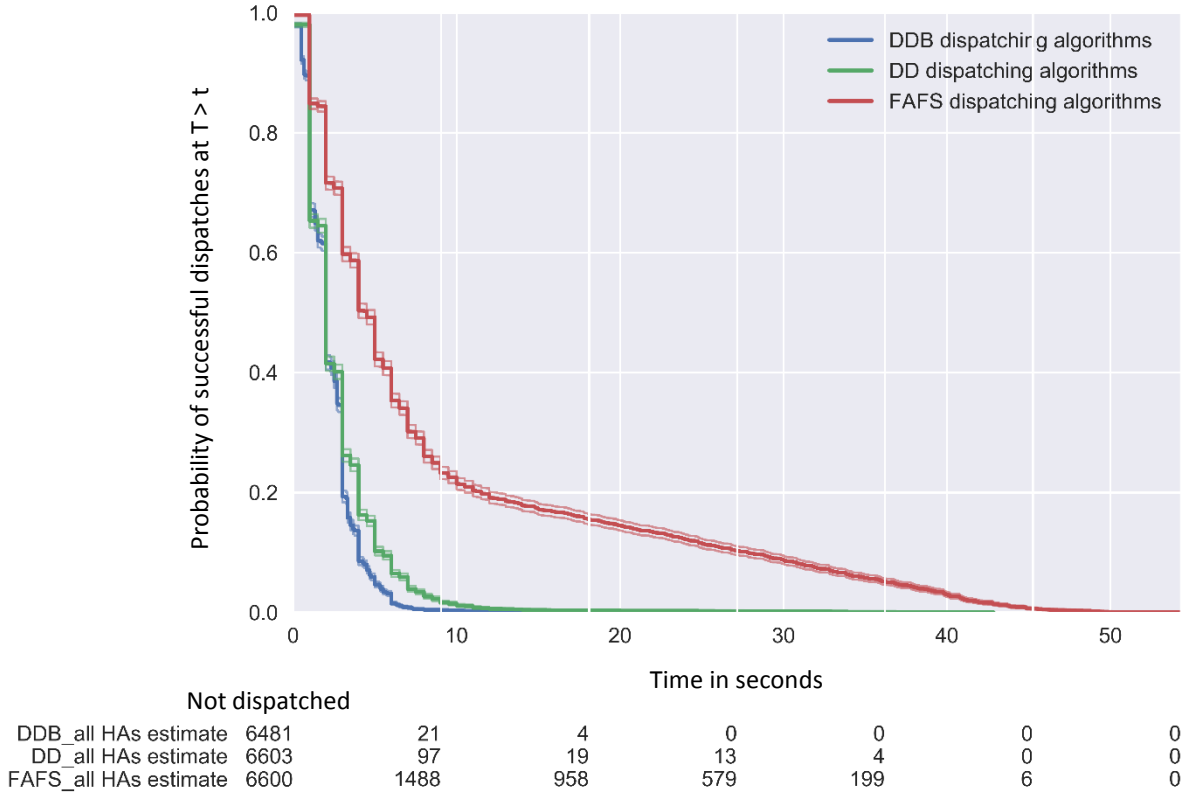


Fig. 6.15. The DDB dispatching algorithm (blue curve), the DD dispatching algorithm (green curve), and the FAFS dispatching algorithm (red curve) reliability analysis estimate.

The DDB dispatching algorithm queueing system was *M/ M/ m* which had a utilization factor The RTAs utilisation factor $\rho = 0.12$. The probability of having an empty request queue $P(0) = 89\ \%$. However, the chance of having three dispatching requests waiting to be served $P(3) = 0.023\%$. It was almost impossible to have all the HPAs waiting to be served since $P(10) = 2.7 \times 10^{-14} \approx 0\ \%$. The number of queueing requests in the queue $L_q = 0.00002$, and the number of requests in the system $L = 0.115$ which was significantly low compared to the DD and FAFS dispatching algorithm.

The waiting time in the request queue $W_q = 4.4 \times 10^{-4}$s which is significantly low and the service waiting time in the system $W = 2.4$s. The total harvest time was 35.9h working days while the total service waiting time for all HPA models was 4.1 h only.

6.6.4 The Heterogeneous RTA Models Utilisation Analysis

The deployment of heterogeneous RTAs had opened up new areas to investigate such as the system overall performance and each RTA performance. Investigating the performance aspects helps answering the following questions:

1. Does the algorithm assign each RTA according to their technical specifications?
2. How often did the algorithm assign the first available RTA regardless of their technical specifications?
3. Which RTA handled more requests and which RTA travelled long distances?
4. Does the heterogeneous multi-agent system perform better than the homogenous system?

These questions are answered by studying the each RTA model behaviour during the DDB dispatching algorithm simulation. The algorithm has three predefined conditions to select the most suitable RTA for each incoming request. For example, the big RTA model handles the nearby requests while the medium RTA model handles requests within a medium distance and the small RTA model serves requests at further distances. However, the algorithm has to respond quickly and not to keep every incoming request waits for a long time until a specific RTA model becomes available.

Another issue to consider is the RTA model utilisation. There is a high chance that the algorithm will not assign the RTA model to serve HPA model out of their service ranges. The HPA movement is unpredictable, and they could skip a tree or two, or group of HPAs may decide to abandon the current row and move to the next one since it has more, or riper fruit. Thus, the big and medium RTAs will not be assigned. Instead, the algorithm will keep assigning the small RTAs. The same condition applies for the smalls RTAs since most of the dispatches are at a small or medium distance.

As a result of these possibilities, the fourth condition was added to overcome the RTAs' utilisation issue. This condition also targeted the RTAs which had not been assigned for a long time. It allowed the system to assign the first available RTA if the RTAs selection conditions were not satisfied. It also forces the RTAs to move closer to the HPAs if they become out of their service range. The system model observed each RTA model's travel time in order to study their behaviour. The Kaplan-Meier estimator was used to study the RTA models travel time results.

The big RTA model's Kaplan-Meier plot estimator is shown in Fig. 6.16. It served 3404 dispatching requests which is 52.2% out of out of 6481 dispatching requests served by the three RTAs. The mean of the service waiting time was 2.09s and the maximum service waiting time was 18.7s. There was a 1% chance to handle an incoming request instantly since $\hat{R}(0) = 0.99$ and only 28 out of 3404 requests were served 0s. The $\hat{R}(3) = 0.063$ indicating that there was a 93.7% chance to handle a request within 3s. The big RTA model handled 3184 dispatching requests within 3s, but 220 requests took more than 3s in which it compensated for the absence of the medium RTA model 209 times and the small RTA model 11 times.

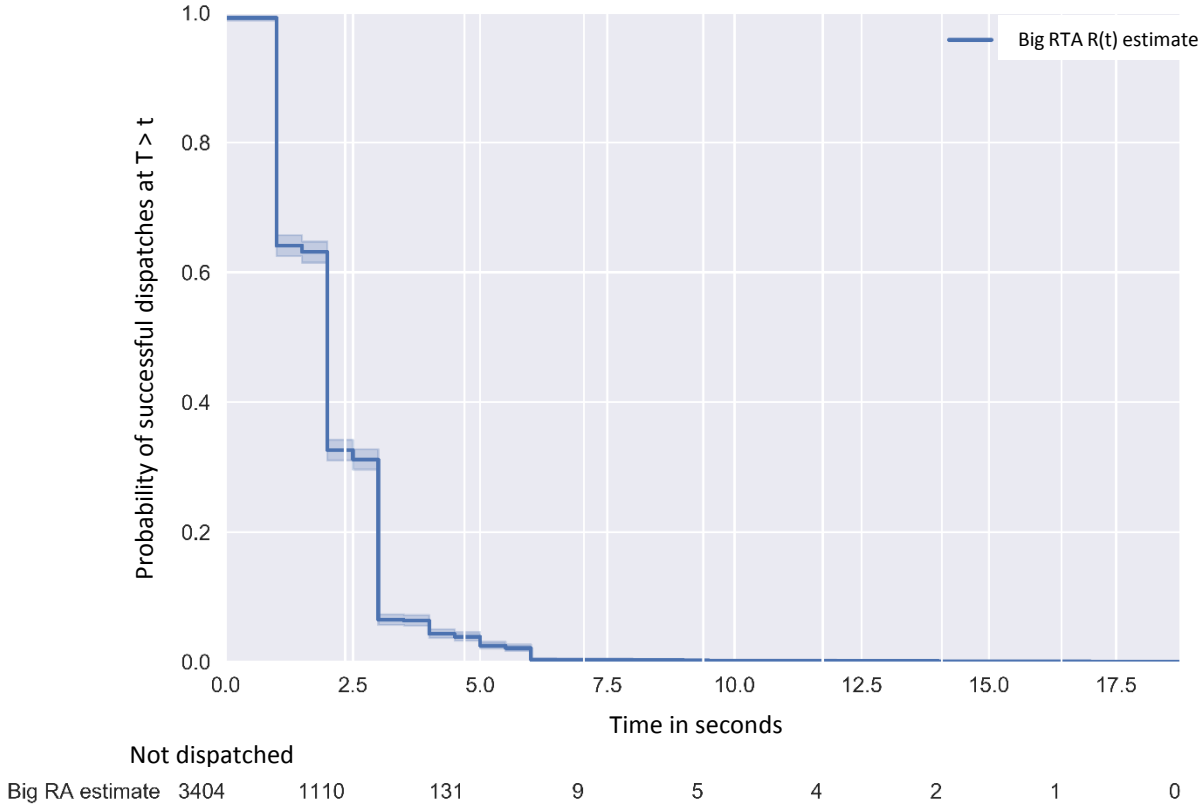


Fig. 6.16. shows the big RTA reliability analysis.

The medium RTA model Kaplan-Meier estimator plot is shown in Fig. 6.17. It served 1467 dispatching requests which is 22.6% out of 6481 dispatching requests. The service waiting time mean and maximum waiting time were 2.9s and 10.5s. The travel time in the estimator plot is the medium RTA actual travel time which is 1.5 times the big RTA travel time used for the DDB algorithm comparison. There were only 48 out of 1467 requests served instantly or $\hat{R}(0) = 0.97$. The medium RTA model had dispatched 618 requests out of 1467 within 3–6s. The model compensated for the absence of big RTA model for 813 times and for the small RTA model 36 times which explains the Kaplan-Meier plot's slow decline from 0 to 3s and the steep decline after 3s.

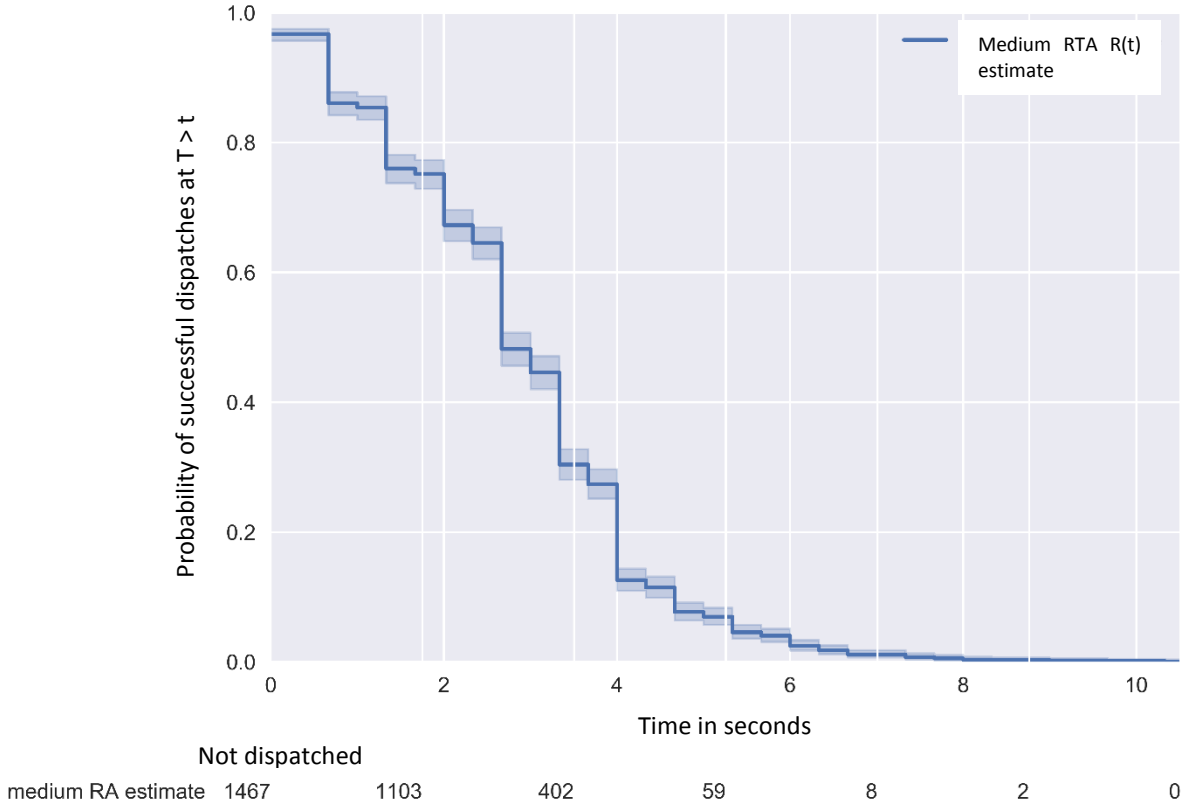


Fig. 6.17. The medium RTA reliability analysis.

The small RTA model Kaplan-Meier estimator plot shown in Fig. 6.18. This analysis is based on the small RTA actual travel time which is half of the travel time used at the algorithms conditions. In other words, one second in Fig. 6.18 is equal to two seconds during the execution of the algorithms condition. The small RTA model handled 1610 dispatching requests which is 24.8% out of 6481 dispatching requests. The mean service waiting time and maximum waiting time were 2.2 s and 21.9 s. The $\hat{R}(0) = 0.96$ since 58 requests served instantly. The dispatching requests which the small RTA took more than 6s to serve was 74 out of 1610. The small RTA model compensated for the absence of the big RTA model 1230 times and the medium RTA model 330 times. The small RTA's speed is 4 m/s twice as fast as the big RTA model.

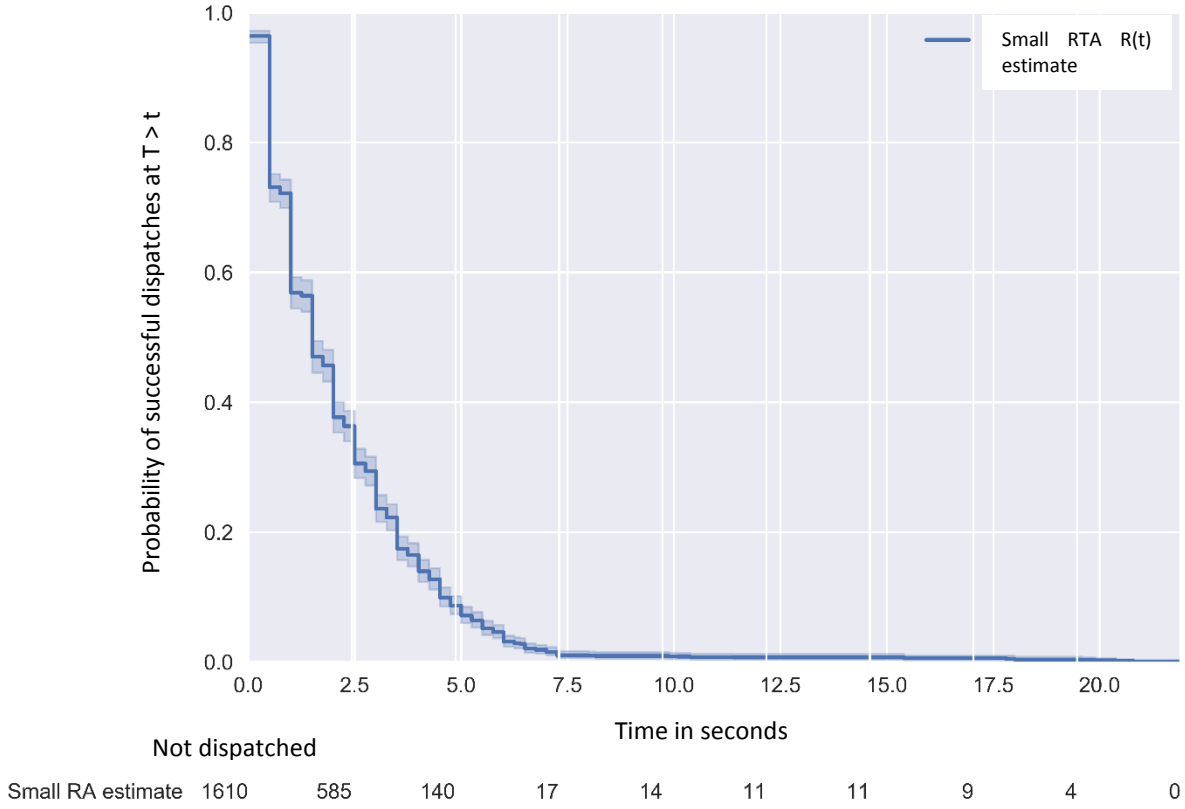


Fig. 6.18. The small RTA reliability analysis.

The small RTA model Kaplan-Meier estimator plot has a larger spread than the big RTA and the medium RTA models since it was only assigned to dispatch requests at a distance, or if both RTAs were busy. Moreover, it has a higher service rate than the medium RTA model, since it was rarely assigned. As a result, the small RTA was available most of the time. However, the algorithm's fourth condition allowed the system to assign the small RTA whenever the three selection conditions were not satisfied to enhance its utilization.

6.7 Cost Optimization

6.7.1 Cost Optimization experiment setup

This experiment investigates the optimal system configuration for deploying heterogeneous RTAs. The cost is analysed based on service waiting time and the type of deployed RTAs. Four

scenarios were simulated while each scenario was Monte Carlo with 50 runs and deployed three RTA models. The RTA models and the simulation results are shown in Table 6.6.

The RTAs operating cost which includes the depreciation cost plus the running cost and HPAs wages costs were used this analysis to estimate the cost waiting time. Based on Section 3.1.2, it is assumed that the operating cost of the big RTA is equivalent to a tractor with a lifespan of 10000h [95] which has 57.1 NZD/h running cost plus 8 NZD/h depreciation cost. The cost of medium RTA is assumed to be equivalent to the operation cost of an electric forklift [96] which as has an expected lifespan of 10000h [97] and 5.8 NZD running cost/h plus 2.18 NZD/h depreciation cost [98]. The small RTA operating cost is assumed to be equivalent to an electric all-terrain vehicle with an upfront cost of 2500 NZD [99] and assuming expected lifespan of 1000h of operating cost [100] since the electrical charging cost is 0.10 NZD/h plus the maintenance cost is assumed to be 0.5 NZD/h plus 2.5NZD/h depreciation cost.

6.7.2 Cost optimization Results

The total cost was calculated by Eq. (6.13) where $C_w = 25$ NZD/h is the worker maximum wages as presented in Section 3.1.2 while C_s, and L are shown in Table 6.7.

Table 6.7. The cost calculation of the four simulation senarios.

RTA	Speed m/s	RTA payload in kg	Mean of waiting time in h	Mean of dispatching inter-arrival in s	λ per 1 h	α per 1 h	ρ	L	RTA operating cost in NZD/ h	Total cost in NZD/h
Three small RTAs										
Small	4	200	0.0005	0.0053	188.7	2000	0.09	0.09	3.1	11.6
Small	4	200								
Small	4	200								
Three medium RTAs										
Medium	3	400	0.0006	0.0055	181.8	1666.7	0.1	0.12	7.18	24.5
Medium	3	400								
Medium	3	400								
Three big RTAs										
Big	2	800	0.00086	0.006	166.7	1162.8	0.14	0.14	65.1	198.8
Big	2	800								
Big	2	800								
Three RTAs (Big, Medium, and Small)										
Small	4	200	0.00063	0.0054	185.2	1587.3	0.12	0.12	3.1	78.4
Medium	3	400							7.18	
Big	2	800							65.1	

The results shows that L which is the number of requests waiting the queue is equal to the RTAs utilisation factor ρ in three scenarios since the RTAs had a small utilisation factor and that the detaching requests did not queue for services. It obvious to have small RTAs over big or medium RTAs due to their cheap operating cost, less service waiting time, and low effect on soil. The disadvantage of using small RTAs in agricultural is that they would not be useful for large scale tasks such as bulk transporting or towing.

6.8 Summary

Automating the yield management has improved the harvest time, safety, and cost in simulation. However, automating the process with a single big RTA prolonged the service waiting time and did not resolve the soil damage. The deployment of multiple small RTAs reduced the soil compaction, but prolonged the service waiting time. The RTAs selection algorithm were investigated to improve the service waiting time, reduce the cost, and the soil compaction.

The FAFS dispatching algorithm which selected the RTAs based on their availability had 8.8s mean service waiting time, the maximum waiting time of 54.3s, and RTAs utilisation of 45.7%. However, the DD dispatching algorithm which selected the RTAs based on their availability and location improved the mean service waiting time by 68.1% and the maximum service waiting time by 21% while the RTAs utilisation was 15%. The DD algorithm was also used to analysis the RTAs utilisation by increasing the number of HPA models and concluded that the RTAs utilisation has an exponential growth and should not be over 80%. It was estimated that a one RTA is required for every 8 HPAs or for every 4ha block. However, to maintain a low utilisation factor the number of RTAs should be increased exponential if the increase on the HPAs number caused the utilisation factor to overshoot over 0.8.

The DDB dispatching algorithm for heterogeneous RTAs which prioritised the RTA's selection based on their location and technical aspects improved the mean service waiting time by 73.9% and the maximum service waiting time by 60% when compared to the FAFS algorithm. The DDB algorithm reduced the likelihood of waiting for a service for more than 6s to 1.5% while the RTAs utilisation was 12%.

The heterogeneous RTAs' utilisation was investigated as well. The big RTA served 55.2% of the total requests, the medium RTAs served 22.6%, and the small RTA served 24.8%. The algorithm achieved a good level of RTAs utilisation due to its predefined instructions to assign the first available RTA if an incoming request did not satisfy the RTAs selection conditions while assessing the available RTAs.

The cost analysis was achieved by simulating the DDB algorithms in four different scenarios while the scenario which had 3 small and fast RTA models significantly reduced the operating cost per hour by 94.2%, the service waiting time by 42%, and expected to reduce soil compaction when compared to the 3 big RTA models scenario. Even though previous studies confirmed the soil benefits of using light and fast agrarian vehicles, the effect of small RTAs on soil needs to be investigated. The optimal cost and waiting time was achieved when simulating three small and fast RTA models.

The service waiting time, number of RTAs required, and RTAs utilisation were improved when executing the DD algorithm which selects available RTAs based on their location. On the other hand, the DDB improved the system response further and allowed the deployment of heterogeneous RTAs for particular tasks while maintaining a collaborative behaviour.

CHAPTER 7: DEVELOPING A SMART BAG SYSTEM FOR MONITORING FRUIT PICKING

7.1 Introduction

The model simulation results have validated the benefits of collaborative systems to automate harvest and yield management. This chapter develops and tests some initial prototype technology required to implement the proposed system. This technology is a smart picking bag which monitors fruit picking, tracks the HPA, and provides continuous interactive communication between HPA and system.

The smart bag has a GPS module which tracks the HPA location to enable the RTA to locate and navigate in an orchard and serve the HPAs. However, the GPS signal can be unreliable, intermittent, and inaccurate around trees. Thus, each RTA should be equipped with a robust sensing and imaging system for successful autonomous navigation.

The simulation results have proven the feasibility of automating harvest and yield management. Thus, a prototype has to be designed and tested to prove the proposed system concept. This chapter investigates the design and field testing of the most critical aspects of the proposed system which are the cooperative interaction between the agents.

7.2 The Fruit Picking System Work Concept

The fruit picking monitoring and robotics navigation system is made of two independent subsystems as shown in Fig 7.1. The first subsystem which is a smart picking bag carried by an HPA to provide data including the HPA location in real time which is discussed in this chapter. The second subsystem is a robotic visual navigation system which helps the RTA to safely navigate to the location provided by the smart bag which is discussed in Chapter 8.

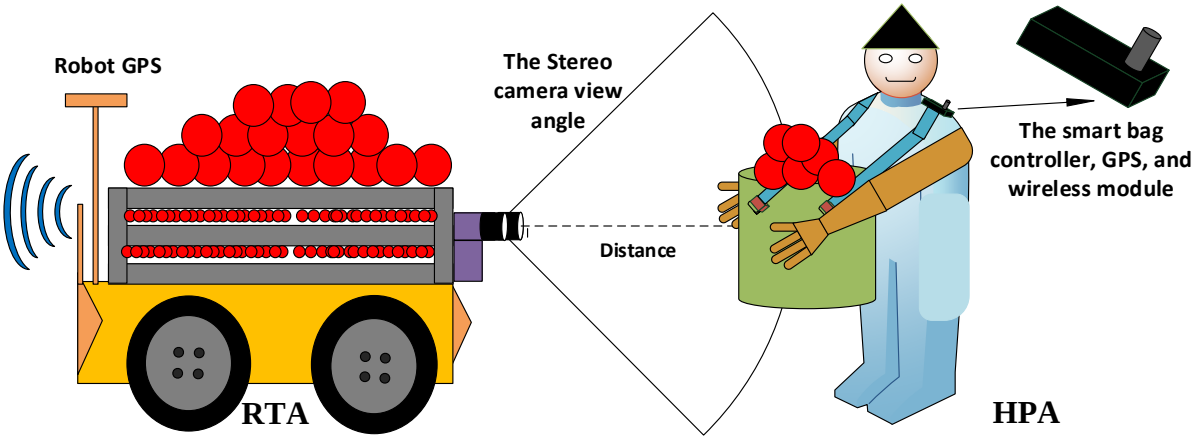


Fig. 7.1. The apple picking monitoring, to the right, and robotics visual navigation system to the left which is discussed in Chapter 8.

The smart bag system monitors the fruit picking process, tracks the HPA location, makes dispatching requests, and interacts with the central controller and the RTAs. The robotic visual navigation and detection system enhances the RTAs navigation around the orchard by visually detecting the HPAs and calculate the distance to safely approach them. The RTAs are serving the HPAs, and they need to safely navigate. The RTAs need to be able to detect, interact with, and operate around humans.

The central controller receives the data transmitted by the smart picking bag and the RTAs. The central controller assigns available RTAs to serve HPAs. The assigned RTA navigates to the dispatching request location which was provided by the smart bags GPS tracker. The provided GPS location is not accurate and might change due to the HPA unpredictable movement. Thus, the assigned RTA visually detects the targeted HPA who made the dispatching request while calculating the accurate distance to the HPA.

7.3 Smart Bag System Design Specification

This part discusses the design and development of a smart picking bag to monitor fruit picking and track the HPA. The bag system has Thin-Film-Transistor Liquid Crystal Display (TFT

LCD) capacitive touch screen which allows the HPA to interact with the proposed system. The smart bag system requirements are listed below:

1. Allows the HPAs to set and reset a weight threshold value at any time.
2. Continuously measures the picked fruit weight.
3. Continuously tracks the HPA location.
4. Broadcasts the fruit weight and HPA location data via a wireless connection.
5. Sends yield despatching requests automatically and manually.
6. Periodically transmits data to update to the central controller and the neighbouring RTAs.

The system working concept flow chart is shown in Fig. 7.2. Once the bag system is switched on, it will connect with the central controller and neighbouring RTAs. Once connected, it will prompt the HPA to enter a weight threshold value not more than a safe recommended value. The HPA starts picking apples, and the bag system periodically updates the central controller by sending the picked fruit's weight and HPAs location as data messages. The bag system requests an RTA to dispatch the yield if the picked fruit weight ≥ the pre-set threshold value or when the HPA presses an RTA *request* button. Once the RTA is called, the bag system waits for the central controller acknowledge receiving the request and assign an available RTA, otherwise, it will periodically keep sending location and weight data.

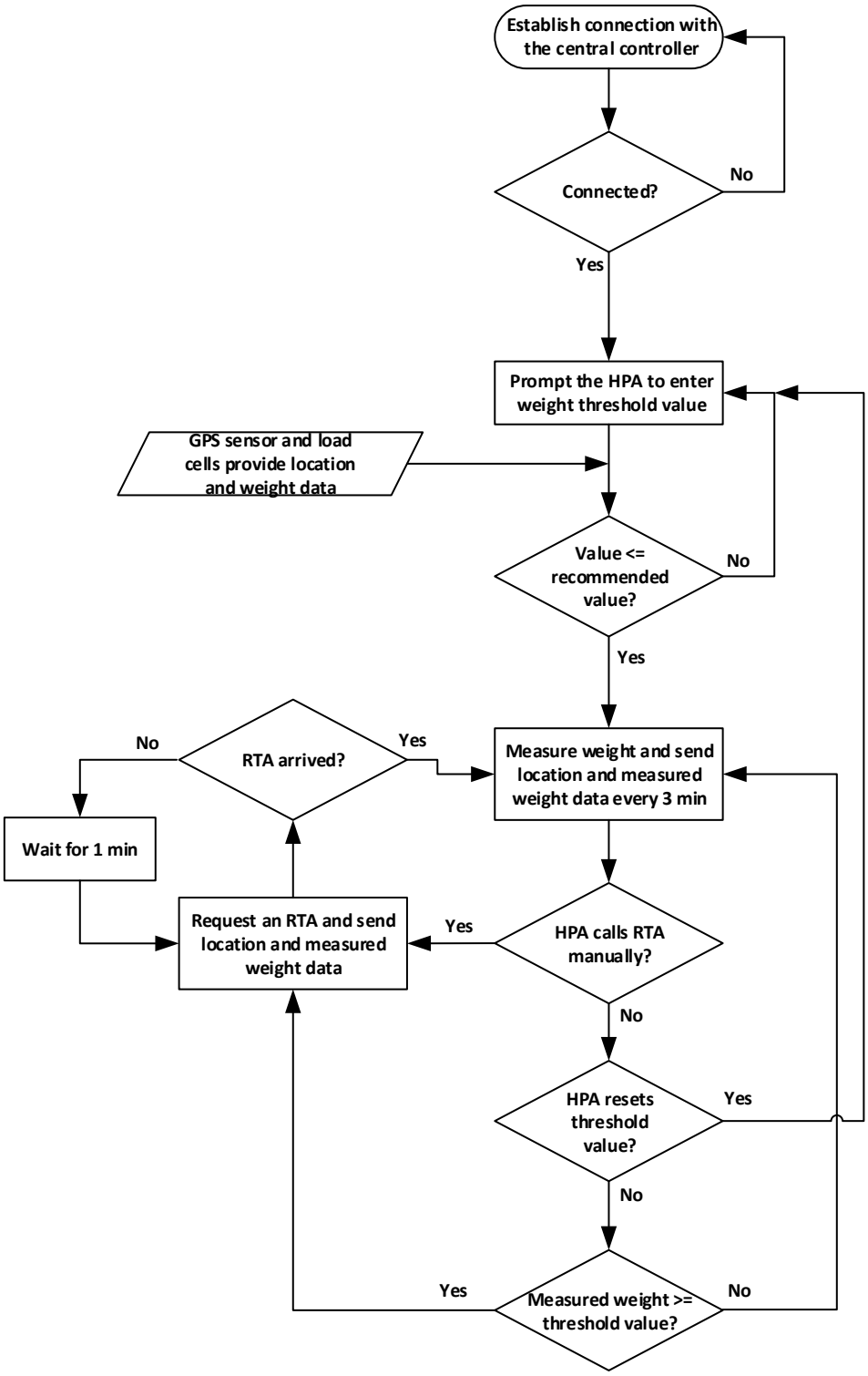


Fig. 7.2. Smart bag control flow chart.

7.4 The Smart Bag Prototype

The smart picking bag prototype consists of a fruit picking bucket, a weight measurement unit, a microcontroller, a GPS location tracking unit, a wireless communication unit, and a

capacitive touch TFT LCD screen as shown Fig.7.3 and the schematic diagram is shown in Fig. 7.4.

Fig. 7.3. Fully assembled smart bag.

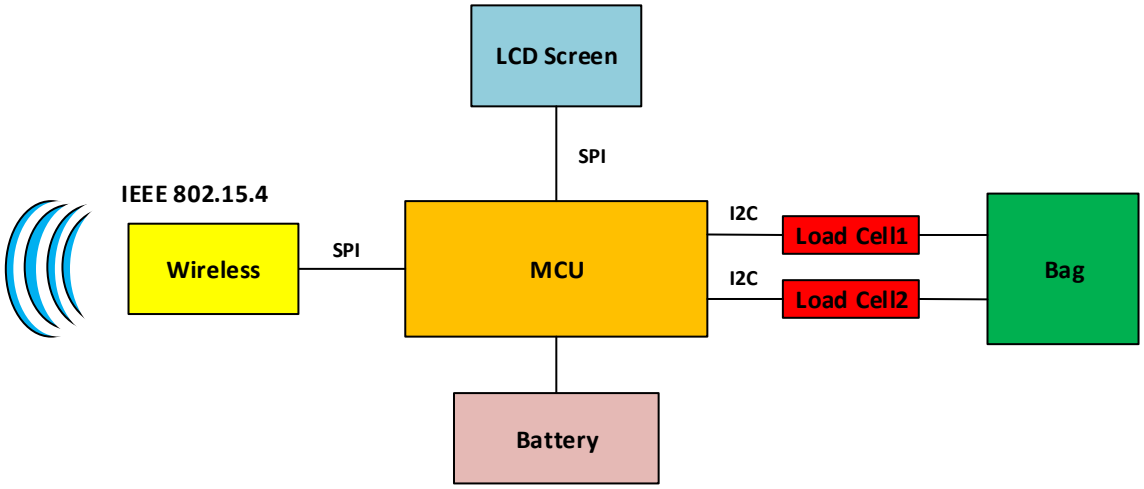


Fig. 7.4. Smart bag sechmatic diagram.

The system components are off-the-shelf and commercially available. The prototype's components and fabrication process are discussed in the following sections:

7.4.1 Fruit Picking Bucket

YIELDMAX fruit picking bucket which was manufactured by Harvestwear Company was used as shown in Fig. 7.5. The bucket has an ergonomic polyethylene kidney-shaped shell, a nylon chute, and two hook catches. The shell is padded with polyurethane foam to reduce fruit bruising. The bucket has an adjustable harness made of strong and comfortable fabric which was shaped to spread over the picker's shoulders and back. The bucket's shape and design make it convenient to pick fruit and walk with the bag.

a) b)

Fig. 7.5. a) YEILDMAX fruit picking bucket (front view), b) Support harness (back view).

7.4.2 Weight Measurement Unit

The weight measurement unit is added to the bucket to measure picked fruit weight in real time. It consists of two load cells model DYMH-103 with a maximum payload capacity of 30kg, two SparkFun HX711 load cell amplifiers, two clevis joints, two rode end bearings, and four rectangle rings as shown in Fig. 7.6(a) while the components are shown in Fig. 7.6(b). The clevis joint and rod end form a precision articulating joint which allows the load cell to retain position during movements and vibrations. The joint vertically mounts the load cells to the bucket shell to absorb angular movements and minimize the measurement fluctuations as shown in Fig. 7.6(c). The rectangle rings were made of an aluminium square bar at the

University of Canterbury's mechanical workshop. They connect the straps to the load cells and the rod ends to the bucket shell.

Fig. 7.6. a) The weight measurement unit, b) The unit's left side components, c) The fully assembled left side.

7.4.3 Microcontroller

The microcontroller which was used in this prototype is an Arduino board MEGA2560. It is based on Atmega 2560 microcontroller and programmed with the Arduino IDE software. The wireless and GPS modules are connected to the board serial communication pins while each load cells are connected to four digital pins. The LCD screen is directly mounted on the controller breakout board as shown in Fig. 7.7. It uses six digital pins and the ICSP headers for the Serial Peripheral Interface (SPI) and Inter-integrated Circuit (I^2C) communication protocols connection. The Arduino board is powered by a 97 x 34 x 8 mm lithium ion battery with a capacity of 2000mAh, which also powers the rest of the system electronic components. The

battery provides 3.7VDC which is boosted by a voltage boost converter to 5VDC. The final system circuit is shown in Fig. 7.8.

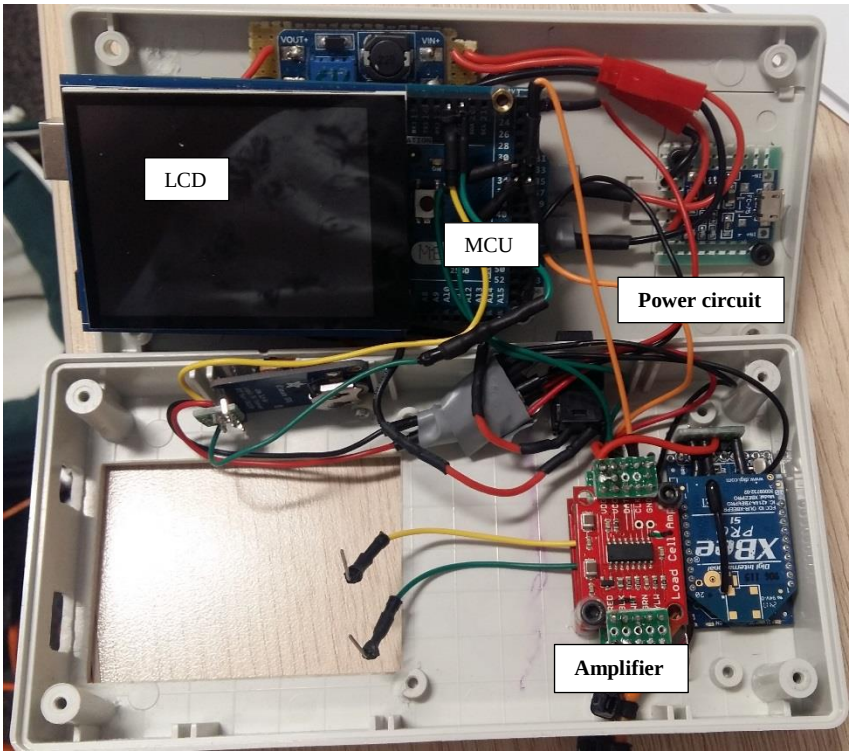


Fig. 7.7. The smart bag circuit including Xbee, amplifies, MCU, LCD, and power circuit.

7.4.4 Tracking and Localization

A GPS module with the model number MTK3339 from Adafruit is used for tracking the HPA while picking fruit as shown in Fig. 7.6. The module can track up to 22 satellites on 66 different channels, and it has a sensitive receiver. It makes 10 readings per second and has a coin cell battery to provide a continuous satellite fix for faster start-up reading. It has the dimensions of 25.5 x 35 x 6.5mm.

7.4.5 Communication

The communication medium used in this proposed collaborative multi-agent system is wireless. The central controller, RTAs, and HPAs broadcast and transmit point to point data messages via a wireless network. Thus, the Xbee-Pro S1 802.15.4 RF module was used in this system since it can provide low latency and multipoint communication. The module device can

communicate in an outdoor line of sight situation up to 1.6 km without an antenna. Two modules are used in this prototype where one of them is connected the smart bag controller while the other is connected to a computer, central controller, via a USB cable as shown in Fig 7.7.

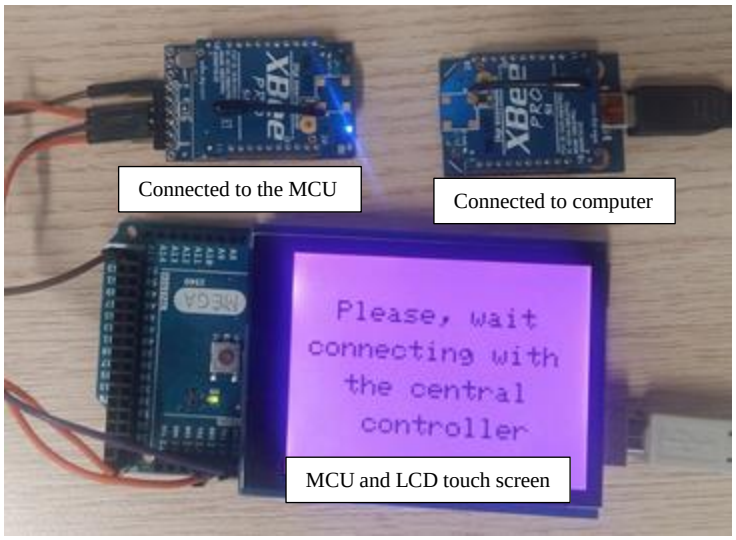


Fig. 7.8. Show two Xbee-pro S1 RF modules and the smart bag trying to connect with the central controller.

There are three types of data messages sent by the bag prototype to the central controller. The first type is a request to establish a connection with the central controller. The second type and third type of messages are periodic data messages which consist of the message type, measured weight, time, and GPS coordinates. The second and third messages types are data buckets which are shown in Table 7.1 and described as follows:

1. A periodic data message type (*P*) sent every 3s
2. A call data message type (*C*) sent every 2s when manually or automatically requesting an RTA.

Table 7.1. Data packets of periodic data messages sent by the smart bag.

Message type	Picked fruit weight	Time	GPS coordinates
P	0.0kg	hh:mm:ss	Latitude, Longitude
C	0.0kg	hh:mm:ss	Latitude, Longitude

The central controller receives and reads the data messages and then transmits back acknowledgement messages. The central controller will send a connection acknowledgement message when receiving a connection message and successfully connected with the smart bag. If the message is a type *P* message, the central controller will store the data into the memory. If the message is a type *C* message, the central controller will send back an acknowledgement message, assign an available RTA, and then send the assigned RTA's ID to the smart bag. If the smart bag did not receive a request acknowledgement message from the central controller, it will keep sending the request message, type *C*, every 2s.

7.4.6 Capacitive Touch TFT LCD Screen

An Adafruit 2.8 in TFT LCD touch shield with a capacitive touch controller was used in this prototype as shown in Fig. 7.4. The LCD allowed the HPA to set a weight threshold value based on their preference which should be less than a safe recommended value. The smart bag sends a dispatching request whenever the picked fruit weight is more than or equal to the threshold value. The HPA can reset the weight threshold value or call for an RTA whenever they want. The TFT LCD screen displays information about when the system is connected, when the picked fruit weight ≥ to the pre-set weight threshold value, and when an RTA is coming.

a) **b)**

Fig. 7.9. a) Welcome message, b) Message to prompt the HPA to key in a weight threshold value.

7.5 Experimental Setup

The field testing was conducted on the 11th of Feb, 2019 in Ilam field, Ilam, Christchurch, New Zealand as shown in Fig. 7.9. Steel machined weights with known mass were used to add weight to the smart bag. There were 8 weights of 0.5kg, 6 weights of 0.75kg, and 6 weights of 1kg. The weights were picked up from the ground and placed into the bags as shown in Fig. 7.9. A laptop was used as the central controller and was connected to the Xbee-pro RF module to communicate with the smart bag.

The central controller implemented an algorithm which reads the data messages transmitted by the smart bag and assigns an available RTA. The central controller assigns the RTA whenever the smart bag automatically or manually sends a dispatching request. The data are stored to a comma-separated value (CSV) file in the central controller for analysis.

Three threshold values of 5kg, 8kg, and 10kg were chosen for this experiment. During the experiment, the HPA walked, picked up weights, put weights inside the smart bag, emptied the smart bag when a virtual RTA arrived and the bag was emptied into it, continued picking, requested an RTA, and reset the threshold value. It is important to note that there was not a physical RTA tested during the experiment and HPAthat the central controller was only assigning a virtual RTA. When the virtual RTA arrives, the HPA emptied the smart bag's contents to the ground simulating that an actual RTA had arrived. For example, a dispatching request was made, the central controller acknowledged receiving the request, informs the smart bag that RTA1 is assigned, and then the HPA would kneel down and empty the bag's contents to the ground.

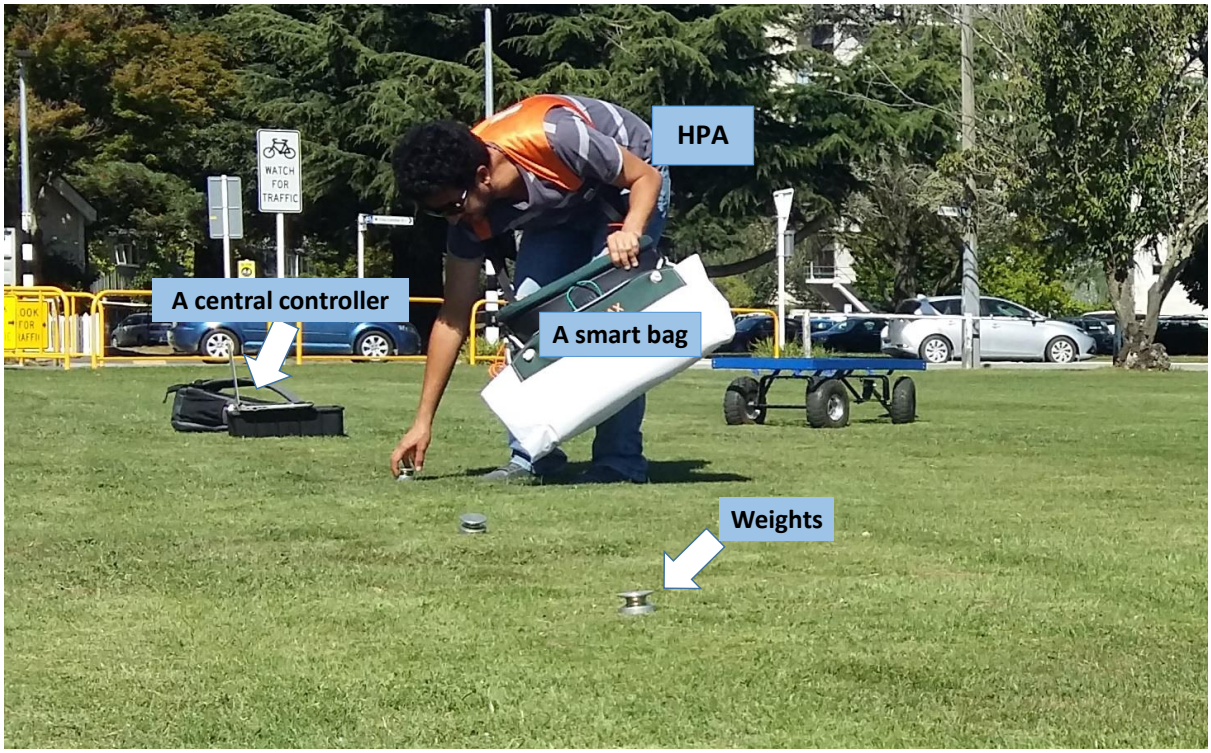


Fig. 7.10. An HPA picking up weights to a smart bag which sends data to a central controller.

7.6 Results and Discussion

The results of the smart bag testing experiment are listed in Table 7.2. The table has 12 smart bags filled by the HPA and dispatched by a virtual RTA. It shows the bag's ID which was only set in this experiment to identify the bags, pre-set weight threshold value, measured weight, the time to fill the bag, how the RTA was requested, and the dispatching location.

The HPA initially set the weight threshold value to 5kg and started picking up weights from the ground. The HPA filled three bags while the bag called for an RTA every time the measured weight was ≥ the 5kg as represented by the blue plot in Fig. 7.10 while the bags locations are displayed in blue pins in the Google Earth map shown in Fig. 7.11. The HPA reset the threshold weight value to 8kg and filled three bags which their weight measurements are illustrated by the yellow plot in Fig. 7.10 while their locations are displayed by the yellow pins in Fig. 7.11. Finally, the HPA reset the threshold weight to 10kg and filled 6 bags which are shown in the red plot in Fig. 7.10 while their locations are shown by the red pins in Fig. 7.11.

Table 7.2. The smart bag experiment data.

Smart bag	Pre-set threshold value (kg)	Measured weight (kg)	Time to fill (s)	Who requested an RTA	GPS coordinates Latitude, longitude (deg : N, t)
Bag1_5kg	5	5.32	120	The bag	-43.5216, 172.5789
Bag2_5kg	5	5.08	94	The bag	-43.5217, 172.5789
Bag3_5kg	5	5.11	190	The bag	-43.5217, 172.579
Bag1_8kg	8	8.09	116	The bag	-43.5218, 172.5789
Bag2_8kg	8	8.54	164	The bag	-43.5216, 172.5789
Bag3_8kg	8	9.25	128	The bag	-43.5218, 172.579
Bag1_10kg	10	10.42	129	The bag	-43.5217, 172.5789
Bag2_10kg	10	8.82	97	The HPA	-43.5217, 172.579
Bag3_10kg	10	10.23	162	The bag	-43.5217, 172.5789
Bag4_10kg	10	4.28	134	The HPA	-43.5217, 172.579
Bag5_10kg	10	10.27	157	The bag	-43.5217, 172.5789
Bag6_10kg	10	4.72	21	The HPA	-43.5217, 172.5788

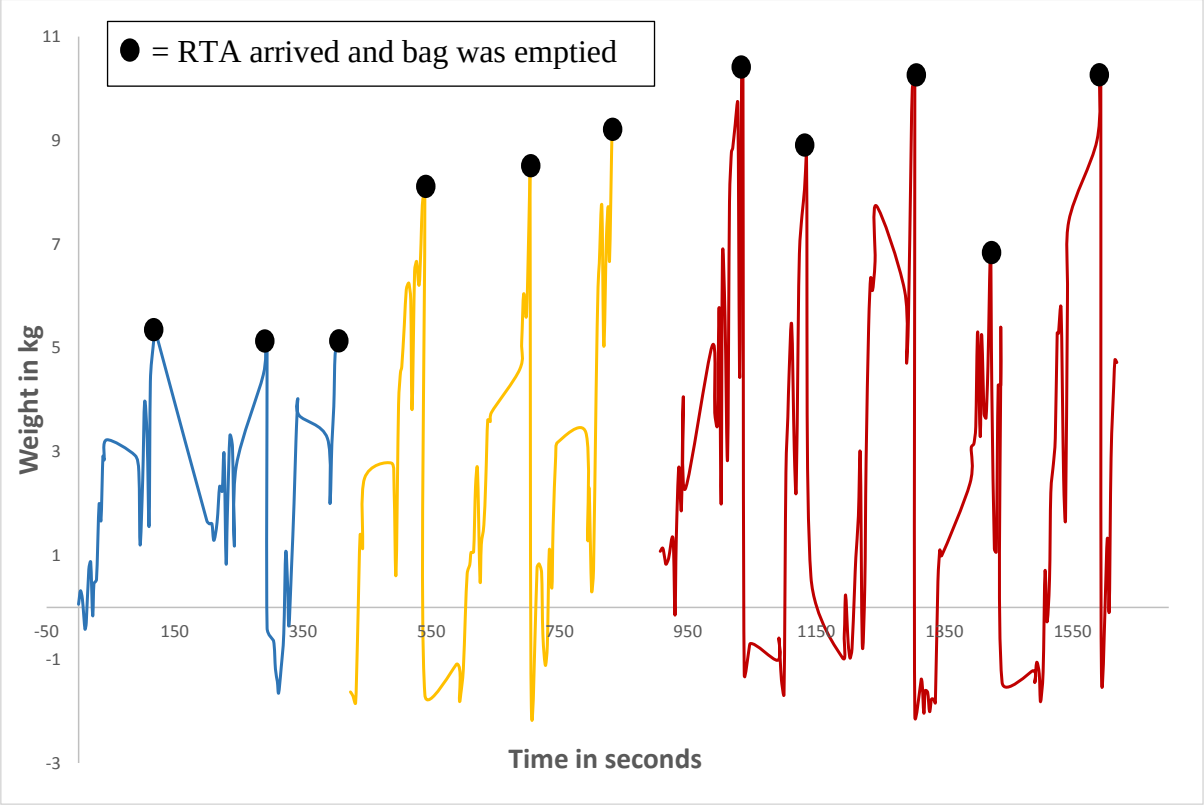


Fig. 7.11. The smart bag weight measurement data, the blue plot represents the three 5kg bags, the yellow plot represents the three 8kg bags, and the red plot represents the six 10kg bags.

Fig. 7.12. The bags' dispatching locations as displayed by dropped pins in Google Earth.

In real time, the smart bag was able to measure the weight, communicate with the central controller, automatically request a virtual RTA, and allow the HPA to manually request a virtual RTA. The bag average the 5 reading from each load cell and then sum the averages to provide the bag weight in time. Five readings were averaged to provide relatively accurate reading. The bag automatically requested a virtual RTA 8 times while the HPA tried the manual request method and called the virtual RTA 3 times. The HPA initially set the weight threshold value to 5kg and filled three bags; later, the HPA reset the threshold value to 8kg and filled another three bags; and finally, he reset the threshold value to 10kg and filled six bags.

The weight measurement results are shown in Fig. 7.10 has some measurement fluctuations due to HPA walking and bending down to pick up the weights. The negative weight values are the results of being down and putting the bag on the ground to empty the bag. The load cells readings are zeroed automatically at the start up of the smart bag system which takes into account the self-weight of the bags. Thus, the smart bag system read negative values when put on the ground, and self-weight is removed.

7.7 Summary

The agent-based cooperative interaction is an essential aspect of the proposed system to be prototyped and tested. This chapter developed and tested a non-existing technology to prove the proposed system concept. A smart bag system was required to monitored fruit picking, enabled cooperative human-machine interaction, and tracked the HPA. The initial prototype is a smart bag which was assembled from off-shelves components to test the proposed system concept.

During the testing of smart bag prototype, it made 11 dispatching requests and each request sent a data message which contain the request location, time, picked fruit weight. These data messages were received by the central controller which assigned virtual RTAs to serve each request. The prototype automatically requested the RTA 8 times while the HPA requested an RTA 3 times.

Overall, the smart bag successfully provided a real-time monitoring of the fruit picking process and cooperative interaction between the HPA, the central controller, and the virtual RTA. The smart bag measurement and localization accuracy can be investigated in future studies.

CHAPTER 8: IMPLEMENTING A VISUAL DETECTION AND NAVIGATION SYSTEM TO ENHANCE ROBOTIC NAVIGATION IN ORCHARDS

It is essential for the RTAs to successfully and safely navigate and approach the HPAs to serve. This Chapter investigates the possibility of applying a robotic visual detection system with open-source detection algorithms to detect the HPAs and enhances the RTAs navigation in orchard environment since it might have intermittent GPS connection. This initial prototype was develop from off shelf open source algorithm to prove the concept of visual detection and navigation can be implemented in orchards. Thus, this technology need to be investigated further in the future.

Robotics navigation systems are based on sensors such as GPS, sonar, vision, laser, telemetry, odometry, or inertia while each sensor has its advantages and limitations [101]. The sensor fusion of different sensors provides reliable measurements for absolute navigation. Moreover, the software and algorithms had improved navigation and enhanced errors compensations.

The current technical development of software, complex algorithms, and hardware have improved sensor fusion including vision sensors. Smart navigation algorithms can predict and filter errors while visually recognise and classify patterns or objects.

This chapter uses open-source visual detection algorithms to aid the RTAs to detect their surroundings and navigate in orchards. This method will allow the RTAs to visually detect, locate, and safely approach the HPAs who made the dispatching requests.

8.1 System Design

The detection algorithm is based on extracting features from images using Histogram of Oriented Gradients (HOG) while measuring the depth of detected objects within the image. The measured depth is the actual distance to a detected object. Thus, a stereo cameras was used

to feed images and depth data. The HOG is used since it is a feature descriptor of objects detection. Moreover, it has a reasonable computational cost, and has become widely used for human detection after being applied by Navneet Dalal and Bill Triggs to detect pedestrians [102].

The HOG detectors which detect the HPAs and their poses were obtained from OpenCV package. The detectors were trained by a training image set and tested on videos which were recorded on October, 2017 at the *Van Herel* apple orchard in Christchurch, New Zealand. The whole set allowed the 3D stereo camera to execute real-time objects detection and distance measurement. The HOG results were compared with a pre-trained Deep Neural Network (DNN) model which was also investigated for visual detection.

HPA's visual detection and the distance measurement flowchart is shown in Fig. 7.12. While the RTA is moving toward the target location the visual system detects the surroundings and measures the distance to objects within the camera's view angle. The RTA will detect trees and obstacle to avoid them. If the camera detected an HPA when approaching the target location, it will check if it is the HPA who made the request. The algorithm detectors are trained to differentiate between the HPA who made the request from the one who did not make the request based on their pose. The HPA who made the request would be standing and waiting for the RTA while other HPAs would be bending down or busy picking apples. Finally, the RTA approached the correct HPA.

8.2 The System Apparatus

The system is made of a stereo camera, CPU, and supporting software. The camera is a ZED Stereo Camera manufactured by Stereolabs Inc. It has two Red Green Blue (RGB) cameras which have a fixed distance between their lenses. The distance between the camera and objects is similar to the depth perception in human eyes.

The ZED camera required NVIDIA GetForce GTX 650 graphics card and NVIDIA CUDA toolkit installed on the RTA controller. The Open Source Computer Vision library (OpenCV) used to process images provided image processing algorithms. It supports the GPU with NVIDIA CUDA. Table 8.1 shows a list of equipment used in this visual detection system.

Table 8.1. equipment used for robotics visual navigation.

Name	**Version compatible with HOG**	**Version compatible with DNN**
PC-Hardware		
Intel processor	Core i7-3770	
NIVIDIA GeForce	GTX 650	
Operating system		
Linux Ubuntu	16.04 LTS, 64-bit	
Software		
ROS	Kinetic Kame	-
ZED SDK	2.1.2	-
NVIDIA CUDA toolkit	8.0	
OpenCV	3.1.0	3.3.1
CMake	3.5.1	

8.3 Objects Detection Methods and Algorithms

The local appearance of objects in a picture is dependent on the intensity change referred to as gradients. The gradients' magnitude changes abruptly at the edge of objects in a picture. The change on gradients enables shapes' feature extraction from a 2D image as seen in Fig. 8.2 which shows a picture taken at the University of Canterbury study room

a) b)

Fig. 8.1. a) A Manually cropped image for the HOG. b) Magnitude of gradient.

The gradient magnitude $|\vec{g}|$ and direction $\emptyset$ are calculated by Eq. (8.1) and Eq. (8.2) given that g_x and g_y can be calculated from the grayscale pixels patches by subtracting the intensity values of neighbouring pixels. OpenCV uses Eq. (8.3) to convert the image's colours into grayscale values as shown in Fig. 8.3.

$$|\vec{g}| = \sqrt{{g_x}^2 + {g_y}^2} \tag{8.1}$$

$$\emptyset = arctan\frac{g_y}{g_x} \tag{8.2}$$

$$Grayscale\ value = 0.299 * R + 0.587 * G + 0.114 * B \tag{8.3}$$

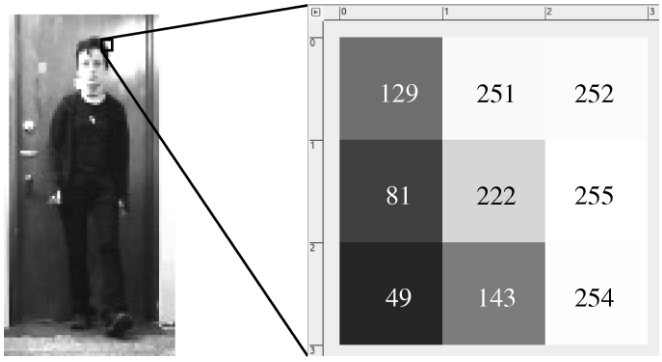


Fig. 8.2. A 3x3 grayscale pixel with intensity values. The picture was taken at the University of Canterbury study room.

The algorithm calculates a feature vector for every image based on HOG bin votes and per normalized 16x16 pixel block. Each image is classified as either positive image which has the object or a negative image which does not have the object. Both positive and negative images vectors are used to train the Support Vector Machine (SVM). The SVM was used since it uses a Support Vector Classification (SVC) and Support Vector Regression (SVR) which calculate a hyperplane which separates positive and negative images vectors. Thus, the calculated hyperplane can easily categorize new vectors which were not introduced within the training set. Moreover, the SVM has the advantage to be trained by recorded data set and tested separately from the camera.

The hyperplane separates data linearly since the vectors' weights are perpendicular to it. The closest vector to the plane is called the support vector which set a classification margin for a robust classification. However, the training data cannot always be separated linearly, and the misclassified data can be taken into account. Data misclassification and margin violation tradeoff are weighted with a parameter *C*, and this type of classification is called C-SVC. Another higher dimension data mapping is achievable for non-linear separable data by applying kernel functions.

The detector is trained by a training set of feature vectors for which the SVM has calculated the hyperplane. The hyperplane can correctly classify vectors of the same dimensions to the training set. Therefore, image resizing is applied buy an algorithm. The algorithm moves a detection window across the whole image, which is referred to as window stride, to detect object that fits the trained size and rescale the image as shown in Fig. 8.4. The HOG detector calculates the resized images vectors that are inputs to the SVM which classifies them as positive or negative images. The detector makes bounding boxes or rectangles on objects in the image and fits their feature vectors to the trained ones and if the specified region matches, it is successful object detection.

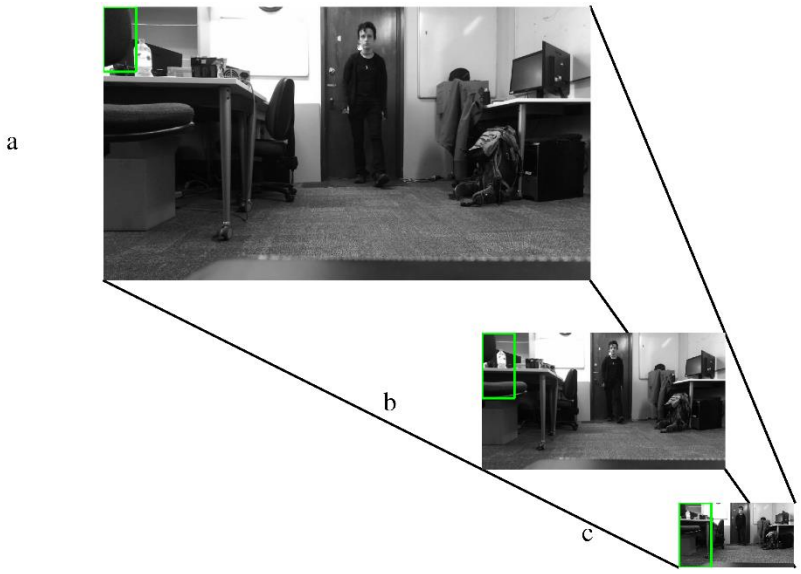


Fig. 8.3. a) Original size image, b) 50% resized image and c) smallest possible resizing. All images has 64x128 pixels detection window in green.

8.4 Model Training

The HPAs stand or bend down while picking. An HPA who filled their picking bag and made a dispatching request are not picking more fruits, thus, they are assumed to be standing and waiting for an RTA. Therefore, the SVM is trained to detect standing and bending down HPAs who carry picking bags. Training the SVM model is achieved either by using the ZED camera, pre-recorded videos, or pictures as seen in Fig. 8.5.

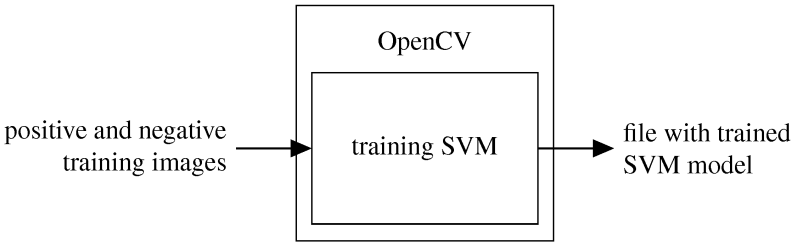


Fig. 8.4. SVM training process.

The training image set which were recorded at a local orchard was prepared before being introduced to the SVM model. A standing HPA image had a detection window size of 64x128 pixels which were recommended by [102]. While the bending down HPA image has a window

size of 80x80 pixels which is still suitable for the calculation presented in Section 8.3. The bending down image was resized by cropping the original image to remove the irrelevant features as shown in Fig. 8.6. The images were cropped initially and then an algorithm was developed to crop and resize the new images. The training set had 678 images for standing HPA, 50 images for bending down HPA, and 568 negative images which did not have standing or bending HPA as presented in Appendix E1.

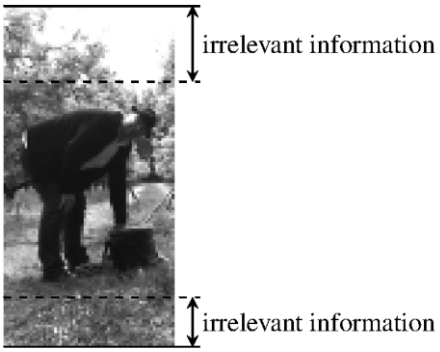


Fig. 8.5. Standard 60x128 pixels size for bending down HPA.

The training model prepares, loads, and saves positive images with a label (+1) and then process the negative images in the same manner with a label (-1). Nevertheless, the original non labelled positive and negative images are still used again for a retaining process. The algorithms convert every labelled image to grayscale values and calculate their feature vector. The SVM model retrains itself while testing new negative images. The maximum number of model retraining is predefined.

This SVM model has multiple detectors which detect the HPAs pose while picking apples. The algorithms initialize each detector based on predefined parameters and execute groups of detectors or single detectors such as object detection, detection filtration, and distance acquisition.

The HOG system starts by loading the trained SVM model and add detectors. Later, it starts the camera and runs the detection process. The detectors grab and retrieve images frame for

positive detection. The detection filtration process eliminates nested, similar, and repeated detections. A detector bounding box is shown on the image frame for every successful detection.

The SVM model was trained with an SVR with linear kernel as suggested by [103] in order to separate data linearly by using a Kernel function to map them to a higher dimension. However, the linear kernel SVR's hyperplane classifier requires pre-setting a *C* and *P* parameters. The *C* parameter which was the hyperplane weighting outliers' multiplier was fixed to 0.01 as suggested by [102] while the *P* which was the OpenCV proportional value to set the margin regression was investigated. The training process was affected by the number of positive images, negative images, *C* value, *P* value, the images scale, and the window stride. The detectors were tested on the verification sample which is presented in Appendix E2.

8.4.1 Standing HPA Detector

The *standing HPA detector* was trained by 678 positive images which captured standing HPA on an orchard as seen in Appendix E1. On the other hand, the best number of negative images to train the *standing HPA detector* was investigated. The detector detects HPAs who are standing and walking away from and toward the camera. The best method is to have a large number of negative images, but a good ratio between the negative and positive images is necessary to set the best *P* value and reduce wrong detections.

The required images scale and window stride depend on the image original size. The scale is needed to compute the image resizing or image pyramid and to get many layers of detection as shown in Fig. 8.4. The window stride is required to determine the number of pixels which the detection window move across. The testing results of different scale and window stride values are shown in Table 8.2. The best values are the scale of 1.1 and window stride of 4x4 pixels which were set for the standing people detection.

Table 8.2. Results of investigating and modifying scale and window stride parameters for *standing HPA detector*

Parameters		Detections		
Scale	**Windows stride**	**Correct**		**False**
1.01	8x8 pixels	32.8%	19	18
	4x4 pixels	41.4%	24	58
1.05	8x8 pixels	60.3%	35	1
	4x4 pixels	72.4%	42	6
1.1	8x8 pixels	50%	29	2
	4x4 pixels	96%	40	1

The detector retaining iteration was required to maximize the number of negative images. The retaining results which initially set the *P* value to 0.9 showed that the best number of retraining iteration was three times since the number of detected negative samples dropped below 20 after the third iteration. During the first iteration, the SVM produced 40000 to 60000 additional negative samples.

The best number of required negative samples for this detector determined the best *P* value. The *P* value controls the regression's margin which balances the number of negative to positive sample. The number of negative images chosen to investigate the impact of *P* value was 339 since it produced the highest number of correct detections with less false detections as shown in Table 8.3.

Table 8.3. Results for verification sample to determine the best number of negative sample required (* = bending HPAs detection).

Parameters			Detections		
negative images	**resulting samples**	**frames**	**correct**		**false**
339	34201	original	94.8% + 4*	59	1
		mirrored	94.8% + 4*	59	1
339 other set	33620	original	94.8% + 4*	59	5
		mirrored	94.8% + 4*	59	0
566	50395	original	93.1% + 4*	58	0
		mirrored	89.7% + 4*	56	0

The *P* value should be small to reduce the number of wrong detections but without reducing the number of correct detection. Table 8.4 shows the testing of *P* values starting from 0.9 to

0.7. The *P* value of 0.75 was chosen since 96.6% detections plus 4 bending down detections were correct while the mirror detection had 94.8% correct detections plus 4 bending down detections.

Table 8.4. Results for verification sample to determine the best P value (* = bending HPAs detection).

Parameters		Detections		
P	**frames**	**correct**		**false**
0.9	original	94.8% + 4*	59	1
	mirrored	94.8% + 4*	59	1
0.8	original	94.8% + 4*	59	1
	mirrored	96.6% + 4*	60	1
0.75	original	96.6% + 4*	60	1
	mirrored	94.8% + 4*	59	0
0.7	original	84.5% + 4*	53	0
	mirrored	84.5% + 4*	53	0

8.4.2 Bending down HPA Detector

The *bending down HPA detector* parameters were determined after creating a preliminary detector which was trained on the available data set. The preliminary detector was trained by 50 positive images, 5 negative images, *C* value of 0.01, *P* value of 0.9, and three retraining iterations. This detector was used to find the scale and window stride which were set to 1.01 and 8x8 pixels. The results of scale and window stride modification are shown in Table 7.8.

Table 8.5. Results of investigating and modifying scale and window stride parameters for *bending down HPA detector*.

Parameters		Correct detections
Scale	**Window stride**	
1.01	8x8 pixels	2
	4x4 pixels	2
1.05	8x8 pixels	0
	4x4 pixels	1
1.1	8x8 pixels	0
	4x4 pixels	1

The last parameter to set was the *P* value to minimize false detections. The SVM model training was repeated with different *P* values. However, decreasing the *P* value to 0.89 increased false

detections and the trained SVM was not able to provide any more positive detections. Thus, the *P* value was set to 0.9 which was also suitable to detect bending down HPAs.

8.5 Results and Discussion

Table 7.9 summaries the parameters set the HOG model to detect HPAs in an orchard. The model had two detectors to detect the standing HPAs and pending down HPAs. Training the SVM to detect standing and bending down poses was successful. Nevertheless, the main objective was to detect the standing HPAs carrying a fruit picking bag while the pending down HPAs detection was an added feature to enhance the detection robustness.

Table 8.6. Parameters set for HOG detection and training.

Parameters	***Standing HPA detector***	***Bending down HPA detector***
Training		
C	0.01	0.01
P	0.75	0.9
No. of positive images	678	50
No. of negative images	339	15
No. of retraining	3	3
Detection		
Scale	1.1	1.01
Window stride	4x4 pixels	8x8 pixels
Padding	0x0 pixels	0x0 pixels
Detection window size	64x128 pixels	80x80 pixels

It was observed that the positive training samples must balance the negative training samples to achieve the best detection results. Setting the model parameters was important especially the scale parameter. However, the effect scale parameter is less in a moving RTA or moving camera since the sizes of detected objects continuously change as the RTA moves.

The changing of detected objects during RTAs' movement is an advantage to be used for navigation and distance measurement. Moreover, it is not necessary for the RTA to detect every available HPAs on the camera window frame. The RTA can detect HPAs while it is moving.

The RTA initially detected HPA2 in Fig. 8.7(a) while it was moving toward HPA2, it detected HPA1 Fig. 8.7(b). This method allows the RTA to inspect each HPA in the drive row and target the HPA who sent the request.

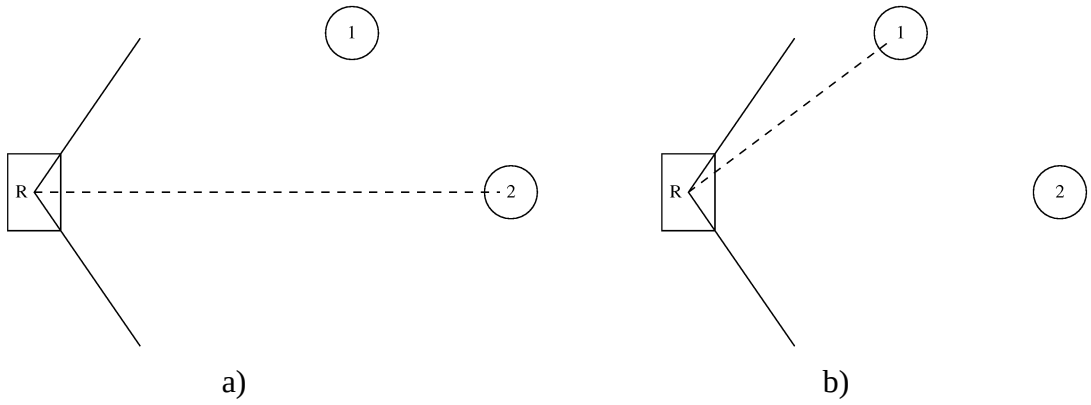


Fig. 8.6. a) RTA detecting and approaching HPA2, b) RTA detects HPA1 while approaching HPA2.

8.5.1 Detection Filtration

The trained SVM model detectors applied software filtration to eliminate false detections. All the detections were saved to a vector and then filtered to remove multiple, equal, and wrong detections. The SVM model has an algorithm that determined wrong detections by matching detection bounding box. The default detectors, not trained detectors, made 60 false detections when tested on the verification samples without applying any filter.

The false detection was improved when filtering overlapping detections. This filter removes the overlapping detections which were bounded over similar bounded detection boxes. This filter reduced false detections from 60 to 55. Another filter was used to filter detections based on their location. The detected HPAs were on ground level, and the detections that aligned above 60% of the image were discarded as they were assumed to be above the horizon. The position filter reduced the wrong detection from 60 to 12. The third filtration method filtered detected objects by their size. The ZED camera can calculate the size of a detected object since it can measure the distance. If the height of a detected HPA within any distance > 0 is less than

1.3 m and more than 2.5 m, it will be discarded. The size detection filter reduced the false detections from 60 to 24.

The HOG algorithms can run multiple filters at the same time to enhance object detection. When applying all the filters to the default detectors, it reduced the false detections from 60 to 3. It is important to mention that the trained detectors were tested on the verification samples and they resulted in 62 correct detections and 29 false detections. When applying the filters to the trained detectors, it reduced the false detection to 0 as shown in Fig. 8.8.

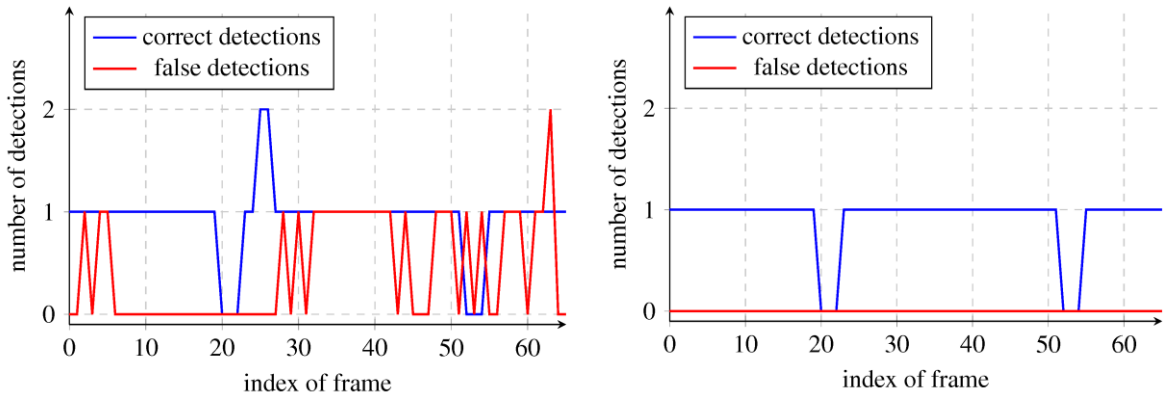


Fig. 8.7. a) Unfiltered detections on trained detectors, b) Filtered detections on trained detectors.

8.6 Distance Detection

The ZED camera can calculate the distance to an object after being detected. A pixel coordinates on the camera's left image corresponds to the point of cloud coordinates (x_{pc}, y_{pc}, z_{pc}) which represent the real world coordinates. The Euclidean distance Z between the camera and real-world coordinates is calculated by Eq. (7.4).

$$Z = \sqrt{{x_{pc}}^2 + {y_{pc}}^2 + {z_{pc}}^2} \tag{8.4}$$

The distance to every pixel in the frame was calculated to include the detected object and its surrounding while the RTA was only interested in the distance to the detected HPA. However,

the detected HPA occupied most of the frame as seen in Fig. 8.9(a) and assuming that a HPA occupies a maximum area of $1m^2$ since the camera only measures depth in full meters and HPA roughly fits 1 m^2. Thus, the distance to every pixel was calculated and then cast off leaving $1m^2$ only as seen in Fig. 8.9(b). The black pixels represented the maximum distance of 20m while the white pixels representing the image edges were set to 0m and the grey colour intensity represented different distances.

a) b)

Fig. 8.8. a) Detected HPA highlighted by a bounding box, b) Calculated distance (white = 0m and black = 20m).

All pixels which had the same distances were counted to create a histogram as shown in Fig. 8.10. The mean of the distribution was 3m, and the 0m pixels were neglected. The detected HPA occupied most of the frame. Thus, the mean value represented the distance to the HPA which would be published to the RTA controller. The RTA approached the detected HPA while evaluating the distance to avoid colliding.

The camera node published calculated distance through ROS framework, and the robot navigation controller subscribes to the data. The results of the published distance on the verification samples are shown in Fig. 8.10 which shows the system detecting HPA within 2 m distance and above since the HPA will be completely visible at such a distance.

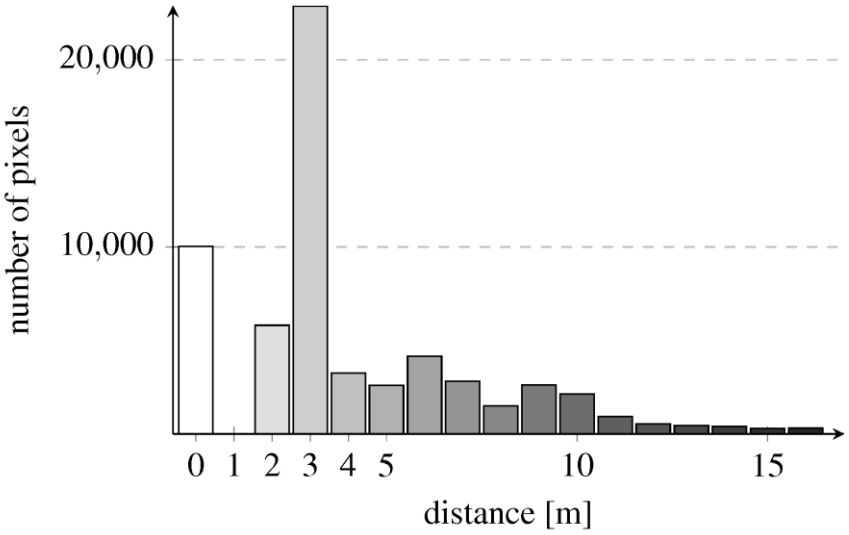


Fig. 8.9 Distribution of measured distances to a detected HPA.

8.6.1 Distance Measurement Accuracy

The distance measurement accuracy was experimented by recoding a HPA waking on an equally divided straight line. A 20 m straight line was divide by visible marks on every meter which an HPA was walking along and toward the camera. The segmented distances with visual marks were used as reference points for the camera to accurately measure the distance. The test was conducted outdoor around trees and the results are shown in Fig. 8.11 which has a distance measurement error of 1%. The error grew, and the wrong distance measurement increased as the HPA gets further away from the camera.

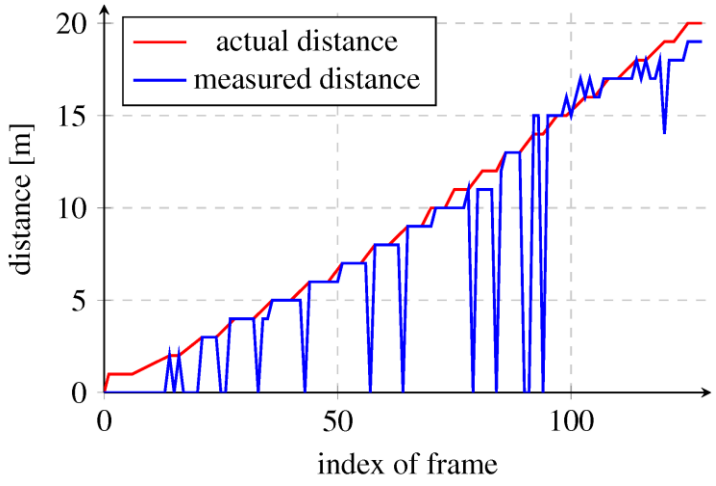


Fig. 8.10. Measured distance to a walking HPA.

The distance measurement errors decreased as the RTA gets closer to the HPA which enhances navigation and also safety. Considering a 1% error over 20 m distance and that the camera only measured the distance in m, the resulted measurement error would be ±1 m. The distribution of correct distance measurement and even the wrong measurements are shown in Fig. 8.12. The results show that the frequency of wrong detection increases as the distance increases.

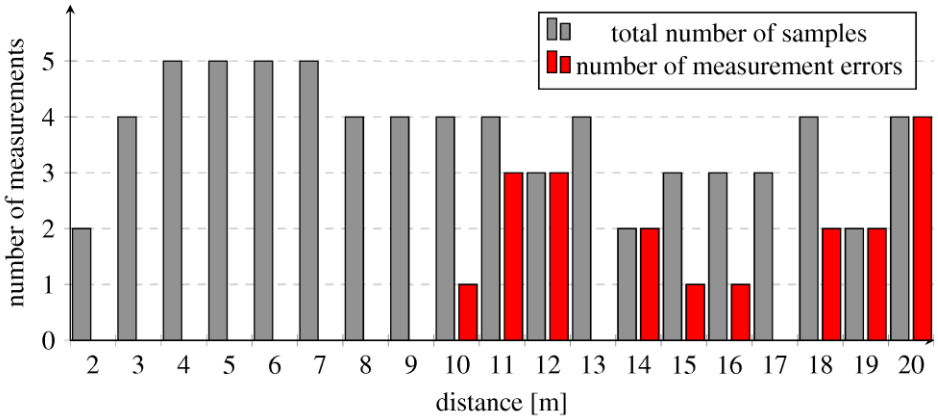


Fig. 8.11 Distribution the measured distances and errors.

8.7 Deep Neural Network (DNN) Detector

The releases of OpenCV after version 3.3 have included a DNN software model. The DNN is a form of artificial neural network (ANN) inspired by biological neural networks. The data flow in one direction and every neuron is connected forming a feedforward network which has an input and output layer with at least one hidden layer between them. A DDN combines its weighted inputs with a bias and transforms the results for the output. The DDN can be trained to adjust its weights and bias and produce the desired output [104]. The DNN model used for HPAs detection parameters which are listed in Table 8.7.

Table 8.7. Parameters for the DNN model for HPA detection.

Parameters	Value	Explanation
DNN model	MobileNetSSD	Open source model for setting up a DNN with parameters
DNN framework	Caffe	Framework used for the DNN model
Input image size	300x300 pixels	Input image size to detect further away HPAs
Scale factor	0.007843f	Factor to scale image channels to filter illumination changes
Mean value	127.5	Mean value that is subscribed from channels to filter illumination changes

The DNN detection model was tested on the verification sample. The model detected 66 HPAs correctly and made 0 false detections without filtration. However, frames 63 and 64 were double detected since the DNN model was also able to detect part of objects and full objects. The DNN model ability to detect part of the targeted objects and partially visible HPAs was an advantage. Thus, two filters were applied which were the overlapping detection filter and the position filter. The filters filter out the double detections as shown in Fig. 8.13.

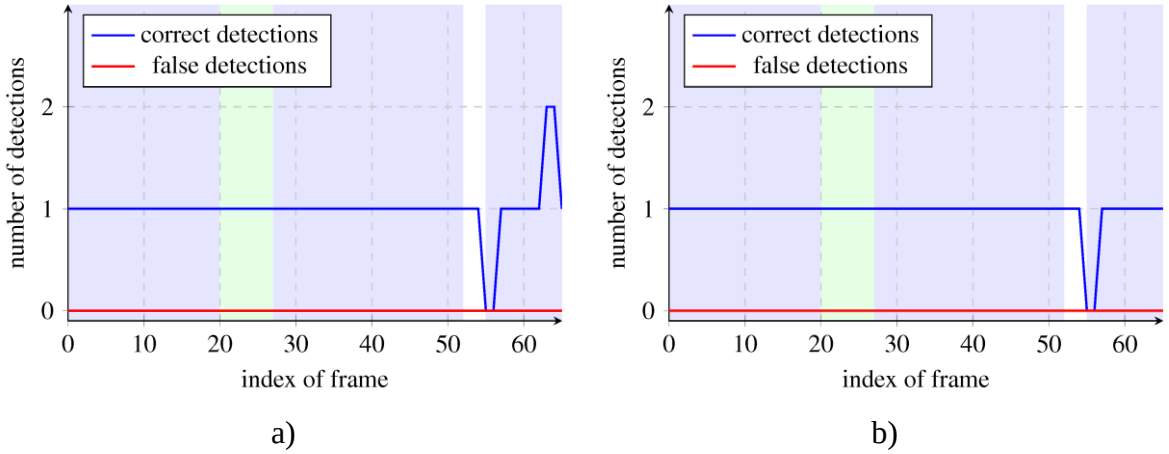
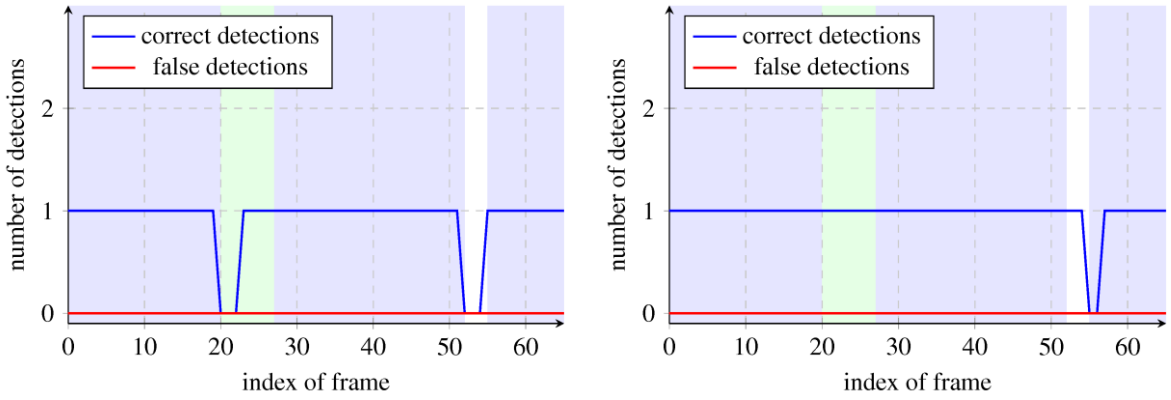


Fig. 8.12. a) DNN model detection without filters, b) DNN detection with filters.

8.8 HOG Model and DNN model Comparison

The HOG model detectors detected only 60 HPAs since it only detected HPAs whose whole silhouette was visible. While applying all filters, the DNN model detector detected 64 HPAs as shown in Fig. 8.14. The DNN model detector can detect partially visible HPA which

improved the number of detects HPAs. However, the DNN model failed to detect upright standing HPA which was visible in frames 55 and 56. Moreover, it can only work on images which have the size of 300x300 pixels. Thus, the images have to be prepared and cropped to fit this specification. However, this can be solved by resizing or implementing a sliding window.



a) b)

Fig. 8.13. a) HOG model results, b) DNN model results.

Both models have successfully detected objects and identify the HPAs who were standing or pending down. The distance to the standing HPA was calculated and published to the RTA controller to safely approach them. The HOG model detection and processing time was 404ms per frame while the DNN model detection and processing time was 46ms. However, the DNN model detection and processing time might increase if frame resizing or sliding window were applied.

Taking everything into account, the DNN model had a better successful detection rate, three times faster, and did not require parameter tuning compared to the HOG model. However, the DNN model missed out some upright pose detections. Both models successfully measured the distance to the HPA and transmitted it to the ROS framework as the RTA controller used to navigate.

8.9 Summary

This chapter discussed an off shelf technology to make an initial robotic vision system prototype which detected the HPAs and enhances the RTAs navigation in an orchard environment. The system allowed the RTAs to visually detect, locate, and safely approach the HPAs.

Two OpenCV detector models were applied which are the HOG and DNN. The detection filtration had to be applied to the HOG detectors to enhance detections. The DNN model was three times faster and had a better detection rate without filters but missed out some upright pose detections. Both models successfully measured the distances to the HPAs which the RTA controller used for navigation.

The developed visual navigation and detection system detected HPAs and accurately measured the distance which was feed to the RTA controller to enhance its navigation in a GPS denied environment. However, this prototype needs to be investigated further to overcome manually tuning the detection parameters for HOG and initial manual images cropping.

The future work is to investigate the DNN model further and to run more tests on actual moving RTAs. More validations are required when developing the proposed system prototype to improve the detection algorithms and create new models based on the DNN since it showed promising results with less processing time. The system needs to try other stereo cameras which have stronger capabilities than the ZED camera or to custom build a stereo camera.

CHAPTER 9: CONCLUSION AND FUTURE WORK

9.1 Conclusion

Previous attempts to automate agricultural tasks have failed to provide technically effective and economically feasible solutions. Growing labour shortages and ever increasing demand for food have inspired the development of the Hybrid Control Collaborative Multi-agent System to Automate Fruit Harvest and Yield management at a minimum cost while enhancing pickers' productivity and safety. This thesis proposes a system architecture for use in fruit-picking scenario and simulates system to validate its utility and behaviour.

A site visit to Plant & Food Research Company's apple orchard in Motueka, New Zealand collected information about apple growing and the harvest process. The site visit has shown that most of the apple farming tasks are highly dependent on labour, especially fruit harvest. An initial review of the literature and interviews with orchard owners identified the possibilities for automation of horticultural tasks, and it was determined that orchards present a large degree of uncertainty which leads to the design of complex automation systems. Previous studies investigated several agricultural automation models and built complex prototypes which had advanced sensors, dynamic algorithms, and powerful processors hoping to overcome the orchards complexity – all of these have proven unreliable and expensive, and industry uptake has been negligible. A key mistake in previous automation attempts is the assumption that the human element non-essential in agriculture. This thesis is novel in considering as an active and cooperative part of the automation system for agricultural fruit harvesting. Having a collaborative multi-agent system allows the human agents to handle the intelligent part of delicately picking fruit, while the robotic transporting agents handle the leg work and manage the yield logistics.

The cooperation behaviour of a multi-agent system enables complicated tasks to be broken down and improves accuracy. Therefore, the use of collaborative systems have improved the robotics perception of human, enhanced their responsive and supportive interactions, and improved safety. Previous systems to automate agricultural activities that aimed to fully automate agricultural activities failed due to being inefficient and costly, but the systems which focused on implementing automation to enhance labour have accomplished acceptable results. By considering the unambiguous benefits of collaborative systems, a reliable multi-agent system can be built to successfully automate harvest and yield management.

A suitable system architecture was proposed. The system consists of human picking agents to handle fruit picking, robotic transporting agents to handle yield dispatching and transporting, and a central controller to execute the processes. The human picking agent picks apples and places them into weight measuring bags which can automatically (or manually) request a robotic transporting agent, which acts as a mobile bin to dispatch the fruit. The system has a hybrid control topology to manage multiple agents' interactions including human-robot and robot-robot. This topology is robust to failures since available robotic transporting agents can compensate for the absence of the central controller and other robotic transporting agents. Overall system reliability is improved by increasing the number of RTAs. The robotic transporting agents have similar reliability as current agrarian tractors with competitive prices. It is estimated that the system could potentially save 58-63% of annual harvesting cost by automating the supervisor and tractor driver roles.

A mathematical model of an apple harvest was developed to simulate a typical apple harvest and yield management method, as well as the proposed automated method. The model has an orchard block model which was represented by undirected and distance weighted graph, human picking agent models, robotic transporting agent models, and a tractor model. The model represented a real apple harvest and simulated the harvest process as a discrete sequence of

events while apple production and fruit picking speed were generated from random uniform distributions. The current method simulated human picking agent models picking apples from trees and walking to fixed bins to empty their picking bags. A tractor model then picks up full bins and transports them to a drop station. The automated method simulates HPA models picking apples from trees to their picking bags which automatically request robotic transporting agent models to dispatch the fruit.

The thesis then simulated the conventional method by deploying three human picking agent models with three different picking experiences but same picking bags' threshold values to harvest an apple block model while a tractor model. Another simulation scenario simulated the automated method with the same human picking agent models and same block model, but it had a single tractor equivalent robotic transporting agent model. The automated method improved the non-harvesting unproductive time by 41.3% and automated the supervisor and tractor driver jobs. However, it had a long service waiting time and did not consider the soil damage. Therefore, a third simulation scenario replaces the tractor equivalent robotic transporting agent model with two smaller size robotic transporting agent models which improved non-harvesting unproductive time by 36.9% when compared to the tractor model scenario. It improved the maximum service waiting by 43.1% and increased the instantly served requests from 278 to 1477 but prolonged the total service waiting time when compared to the single robotic transporting agent scenario. Also, the light robotic transporting agent units decrease soil damage, and remove the need for a tractor operator

The long service waiting time resulted from deploying multiple robotic transporting agent models encouraged the investigation of the robotic transporting agents selection algorithm to improve the service waiting time and the robots utilization. Three algorithms were developed and simulated by deploying 10 human picking agent models which had different picking experiences and picking bag threshold values to be served by three robotic transporting agent

models. First, the First Available First Serves (FAFS) dispatching algorithm which assigns three small size (200kg) robotic transporting agents with speed of 2 m/s based on their availability, had a maximum service waiting time of 0-54.3s with the mean of 8.8s. Second, the Dynamic Distance (DD) dispatching algorithm which also selects the three small robotic transporting agents based on their availability and locations the service waiting time by 21%, the mean by 68.1%, and the robotic transporting agents' utilisation. The proposed system service capacity was investigating using the DD algorithm to determine the relationship between the number of human picking agents and the RTAs utilisation while being served by three small robotic transporting agents. The results showed an exponential relationship between the number of human picking agents and the RTAs utilisation and the number of RTAs. However, it was estimated that a small robotic transporting agent is required for every 8 human picking agents.

Third, the Dynamic Distance and Best fit dispatching algorithm which assigned three heterogeneous robotic transporting agents with different speed and payload capacity based on their availability, location, and speed. The DDB algorithms reduced the mean service waiting time by 73.9% over the FAFS algorithm. The robotic transporting agents and their utilisation results with the DDB algorithm are as follows:

1. The 800kg payload capacity robotic transporting agent with a speed of 2 m/s served 55.2% of the total requests
2. The 400kg capacity payload robotic transporting agent with a speed of 3 m/s served 22.6 % of the total requests
3. The 200kg payload capacity robotic transporting agent with a speed of 4m/s served 24.8 % of the total requests.

The DDB algorithm cost analysis concluded that using small and fast robotic transporting agents is the optimal choice due to their low utilisation and fast service time with the added benefit of reduced soil compaction.

While the simulation results have validated the expected benefits of a collaborative system to successfully automate the apple harvest and yield management, the system requires some technology that is not yet available. Thus, to further test the idea of the agent-based cooperative interaction and the robotic transporting agent navigation in an orchard environment were developed and tested by building a smart bag system and a robotic visual detection system prototype. The smart bag successfully monitored the fruit picking, measured the picked fruit weight, enabled human-robot interaction, tracked the human picking agent, and requested robotic transporting agents for 11 times.

The visual robotic system was able to detect the human picking agents who made the dispatching request and visually navigate to their location in an orchard as a GPS denied environment. The visual system used a stereo 3D camera and two developed HOG and DNN detectors based on an open source OpenCV package. Both detectors successfully detected the human picking agents and the calculated distance to their locations and then fused the distance to the robotic transporting agent controller to approach and serve them. The DNN detector was faster, more reliable, and easier to set up than the HOG detector. However, the method used relied on manual setups, so would not be applicable in its current form.

In summary, this thesis shows a novel approach automation in the agricultural industry by employing a hybrid control collaborative multi-agent system to optimise fruit harvest and yield management. There is an economic and humanitarian need to automate agricultural tasks. The limitations of previous, fully robotic automation designs has led to the necessary development of this alternative system concept that utilises human picking and robotic transporting agents

for tasks they each excel at. The model simulation and field testing showed the potentials of implementing a collaborative technology to automate agricultural tasks. Therefore, the development of such a system brings to future agriculture more than a simple automation system, but also provides a precision agriculture product which collaboratively automates agricultural tasks, provides site-specific data, and enhances efficiency.

9.2 Novelty

This thesis contributes the following:

1. A novel collaborative architecture to automate harvest and yield management. The novelty of this architecture is to incorporate human intelligence with the robotic precision to overcome the complexity outdoor environment
2. A novel discrete-event simulation model to simulate apple harvest which has human picking agents models and robotic transporting agents models cooperatively working together to harvest apples from a virtual orchard block
3. A set of operation and optimisation algorithms which master plan the harvest automation process and manages the optimal service waiting time by selecting the best robotic transporting agent for each dispatching task
4. A novel fruit monitoring technology based on a smart picking bag which measures picked fruit weight, tracks picker's movement, enables human-robot interaction, and collects real-time data for future enhancement.

9.3 Future Work

The system presented in this thesis is a step forward in a long journey of research and development of collaborative systems in the field of agricultural automation. Taking this into account, this research presents several limitations and areas to be explored. These opportunities exist within the system modelling, testing of smart picking bag and visual robotic navigation,

and the whole system prototyping to test and improve its operation and optimisation algorithms.

The system operation algorithms and the optimization algorithms need to be tested on actual robotic transporting agents. There are many aspects to consider when designing a robotic platform to work in an orchard which also depends on its functionality. However, the most important aspects to consider when designing a robotic platform for agricultural application is to minimise the soil damage, to take into account slippery and muddy terrains, and to have a competitive price and better return of investment to current available systems. There is also a need to study the effect of small and fast RTAs on soil.

The smart bag prototype and the visual navigation system were initial prototypes to demonstrate the possibility to implement their technology in harvest. Thus, there is a need to further refine them and analyse their functionally before being tested during an actual apple harvest to collect real data about the pickers' behaviour and the visual detection system behaviour. Although the bag and the visual detector algorithms have considered the worst case scenario, it is still important to make larger scale experimentations and extensive analyses while overcoming the limitation of manually setting up the visual detections.

Simulation with more realistic human factors such as the need for breaks, mistakes, social behaviour, etc. rather than an idealistic situation. As far as future modelling of collaborative systems is concerned, the human model should be developed by taking into account the human picking agents random movement from a tree to another and not simply following a pre-defined path. The new model should consider their movement while climbing ladders and if they decide to temporary leave the block and later resume picking from a different tree in a different row. The current human picking agent model is very simple and generic. Moreover, the robotic transporting agent model needs to be improved by modelling the speed variations and the

random stops caused by obstacles or workers passing in front of the robotic transporting agent. A suggested approach to solve these issues is to consider a finite state machine modelling.

REFERENCES

1. Stringer, C., *Worker exploitation in New Zealand: A troubling landscape.* 2016.
2. Keyser, H. *Agricultural Workers Face Constant, Horrific Human Rights Violations, a New UN Report Shows*. Agriculture 2018 [cited 2018 27/10/2018]; Available from: https://munchies.vice.com/en_us/article/vbkwb3/agricultural-workers-face-constant-horrific-human-rights-violations-a-new-un-report-shows.
3. Arvidsson, J., H. Westlin, T. Keller, and M. Gilbertsson, *Rubber track systems for conventional tractors – Effects on soil compaction and traction.* Soil and Tillage Research, 2011. **117**: p. 103-109.
4. Çarman, K., *Tractor forward velocity and tire load effects on soil compaction.* Journal of Terramechanics, 1994. **31**(1): p. 11-20.
5. Lipiec, J., R. Horn, J. Pietrusiewicz, and A. Siczek, *Effects of soil compaction on root elongation and anatomy of different cereal plant species.* Soil and Tillage Research, 2012. **121**: p. 74-81.
6. Kassler, M., *Agricultural Automation in the new Millennium.* Computers and Electronics in Agriculture, 2001. **30**(1): p. 237-240.
7. Reid, J., J. Schueller, and W. Norris, *Reducing the manufacturing and management costs of tractors and agricultural equipment.* 2003.
8. Shedletsky, A.-K. *When Factories Have A Choice Between Robots And People, It's Best To Start With People*. 2018 [cited 2018 11/07]; Available from: https://www.forbes.com/sites/annashedletsky/2018/06/11/when-factories-have-a-choice-between-robots-and-people-its-best-to-start-with-people/#15a76bed6d5f.
9. Bac, C.W., E.J. van Henten, J. Hemming, and Y. Edan, *Harvesting Robots for High-value Crops: State-of-the-art Review and Challenges Ahead.* Journal of Field Robotics, 2014. **31**(6): p. 888-911.
10. New Zealand Statistics. *Land use*. 2012 [cited 2018 13th of February]; Agricultural Statstics, New Zealand]. Available from: http://archive.stats.govt.nz/browse_for_stats/industry_sectors/agriculture-horticulture-forestry/2012-agricultural-census-tables/land-use.aspx.
11. New Zealand, S. *Fresh fruit and vegetable prices – our global connection*. Stats NZ 2014 January 2014; Available from: http://archive.stats.govt.nz/tools_and_services/newsletters/price-index-news/jan-14-fruit-and-vege.aspx.
12. O'Sullivan, D.L.a.P. *Apples a major player in growth* 2015 15/02/2018]; Available from: http://www.nzherald.co.nz/hawkes-bay-today/rural/news/article.cfm?c_id=1503457&objectid=11489341.
13. Vougioukas, S., Y. Spanomitros, and D. Slaughter. *Dispatching and routing of robotic crop-transport aids for fruit pickers using mixed integer programming*. in *American Society of Agricultural and Biological Engineers Annual International Meeting 2012, July 29, 2012 - August 1, 2012*. 2012. Dallas, TX, United states: American Society of Agricultural and Biological Engineers.

14. Salazar, M.K., M. Keifer, M. Negrete, F. Estrada, and K. Synder, *Occupational Risk Among Orchard Workers: A Descriptive Study.* Family & Community Health, 2005. **28**(3): p. 239-252.

15. Blanes, C., Mellado, M., Ortiz, C., & Valera, A. , *Review. Technologies for robot grippers in pick and place operations for fresh fruits and vegetables.* Spanish Journal of Agricultural Research, 2011. **9**: p. 1130-1141.

16. Barca, J.C. and Y.A. Sekercioglu, *Swarm robotics reviewed.* Robotica, 2013. **31**(03): p. 345-359.

17. Pentjuss, A., A. Zacepins, and A. Gailums. *Improving precision agriculture methods with multiagent systems in Latvian agricultural field*. in *Proceedings of*. 2011.

18. Khalid Salah, X.C., *Cooperative Robotic Systems in Agriculture*, in *Robotics and Mechatronics for Agriculture*, D. Zhang and B. Wei, Editors. 2017, CRC Press, Taylor & Francis Group. p. 132-156.

19. Salah, K., X. Chen, K. Neshatian, and C. Pretty. *A hybrid control multi-agent cooperative system for autonomous bin transport during apple harvest*. in *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 2018.

20. Sebastian, A. *IBM unveils world's first 5nm chip*. 2017 [cited 2019 Jan, 11th]; Available from: https://arstechnica.com/gadgets/2017/06/ibm-5nm-chip/.

21. Aishwarya J. Kurhade, A.M.D., Ruturaj D. Dongare,, *Review on "automation in fruit harvesting"*. International Journal of Latest Trends in Engineering and Technology (IJLTET), 2015. **6**(2).

22. Acevedo, J.J., B.C. Arrue, I. Maza, and A. Ollero, *Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions.* Journal of Intelligent & Robotic Systems, 2013. **70**(1-4): p. 329-345.

23. Peterson, D.L., *DEVELOPMENT OF A HARVEST AID FOR NARROW-INCLINED-TRELLISED TREE-FRUIT CANOPIES.* Applied Engineering in Agriculture, 2005. **21**(5): p. 803.

24. Willrodt, F.L., *Steering attachment for tractors*. 1924, Google Patents.

25. Baerdemaeker, J.D., A. Munack, H. Ramon, and H. Speckmann, *Mechatronic systems, communication, and control in precision agriculture.* IEEE Control Systems, 2001. **21**(5): p. 48-70.

26. Reps, N. and D.L. Schmoldt, *W1009: Integrated Systems Research and Development in Automation and Sensors for Sustainability of Specialty Crops.*

27. Grift, T., Q. Zhang, N. Kondo, and K. Ting, *A review of automation and robotics for the bioindustry.* Journal of Biomechatronics Engineering, 2008. **1**(1): p. 37-54.

28. Robert, P.C., *Precision agriculture: a challenge for crop nutrition management*, in *Progress in Plant Nutrition: Plenary Lectures of the XIV International Plant Nutrition Colloquium: Food security and sustainability of agro-ecosystems through basic and applied research*, W.J. Horst, et al., Editors. 2002, Springer Netherlands: Dordrecht. p. 143-149.

29. McBratney, A., B. Whelan, T. Ancev, and J. Bouma, *Future Directions of Precision Agriculture.* Precision Agriculture, 2005. **6**(1): p. 7-23.

30. Zhang, N., M. Wang, and N. Wang, *Precision agriculture—a worldwide overview.* Computers and Electronics in Agriculture, 2002. **36**(2): p. 113-132.

31. Aubert, B.A., A. Schroeder, and J. Grimaudo, *IT as enabler of sustainable farming: An empirical analysis of farmers' adoption decision of precision agriculture technology.* Decision Support Systems, 2012. **54**(1): p. 510-520.

32. Gebbers, R. and V.I. Adamchuk, *Precision Agriculture and Food Security.* Science, 2010. **327**(5967): p. 828-831.

33. Ross Galbreath. *Agricultural and horticultural research*. 2008 [cited 2018 14 February]; Available from: https://teara.govt.nz/en/agricultural-and-horticultural-research/print.

34. Sánchez-Hermosilla, J., F. Rodríguez, R. González, J.L. Guzmán, and M. Berenguel, *A mechatronic description of an autonomous mobile robot for agricultural tasks in greenhouses*, in *Mobile Robots Navigation*. 2010, InTech.

35. Reynoldson, L., W. Humphries, S. Speelman, E.W. McComas, and W. Youngman, *Utilization and cost of power on corn belt farms*. 1933, United States Department of Agriculture, Economic Research Service.

36. Suprem, A., N. Mahalik, and K. Kim, *A review on application of technology systems, standards and interfaces for agriculture and food sector.* Computer Standards & Interfaces, 2013. **35**(4): p. 355-364.

37. Ko, M.H., B.S. Ryuh, K.C. Kim, A. Suprem, and N.P. Mahalik, *Autonomous Greenhouse Mobile Robot Driving Strategies From System Integration Perspective: Review and Application.* IEEE/ASME Transactions on Mechatronics, 2015. **20**(4): p. 1705-1716.

38. Morgan, K., *A step towards an automatic tractor.* Farm mech, 1958. **10**(13): p. 440-441.

39. Li, M., K. Imou, K. Wakabayashi, and S. Yokoyama, *Review of research on agricultural vehicle autonomous guidance.* International Journal of Agricultural and Biological Engineering, 2009. **2**(3): p. 1-16.

40. Nguyen, T.T., K. Vandevoorde, E. Kayacan, J. De Baerdemaeker, and W. Saeys. *Apple detection algorithm for robotic harvesting using a rgb-d camera*. in *International Conference of Agricultural Engineering, Zurich, Switzerland*. 2014.

41. Mousazadeh, H., *A technical review on navigation systems of agricultural autonomous off-road vehicles.* Journal of Terramechanics, 2013. **50**(3): p. 211-232.

42. Tamaki, K., Y. Nagasaka, K. Nishiwaki, M. Saito, Y. Kikuchi, and K. Motobayashi, *A Robot System for Paddy Field Farming in Japan.* IFAC Proceedings Volumes, 2013. **46**(18): p. 143-147.

43. Flemmer, R.C. and C.L. Flemmer, *Innovations in fruit packing: a slow kiwifruit packing line and a robotic apple packer.* International Journal of Postharvest Technology and Innovation, 2011. **2**(2): p. 120-130.

44. Athukorala, A., N. Ranasinghe, K. Herath, P. Jayasekara, and T.D. Lalitharatne. *Scalable Autonomous Agronomical Smartbot*. in *2018 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA)*. 2018. IEEE.

45. Mitsuhashi, T., A. Yamazaki, and T. Shichishima. *Automation of plant factory*. in *Proc. 4th SHITA symposium, Tokyo, Japan*. 1994.

46. Dos Santos, F.N., H. Sobreira, D. Campos, R. Morais, A.P. Moreira, and O. Contente. *Towards a reliable monitoring robot for mountain vineyards*. in *Proceedings - 2015 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2015*. 2015.

47. Hemming, J., C. Bac, B. van Tuijl, R. Barth, J. Bontsema, E. Pekkeriet, and E. Van Henten, *A robot for harvesting sweet-pepper in greenhouses.* 2014.

48. Scarfe, A.J., R.C. Flemmer, H.H. Bakker, and C.L. Flemmer. *Development of an autonomous kiwifruit picking robot*. in *2009 4th International Conference on Autonomous Robots and Agents*. 2009.

49. Baeten, J., K. Donné, S. Boedrij, W. Beckers, and E. Claesen. *Autonomous fruit picking machine: A robotic apple harvester*. in *Field and service robotics*. 2008. Springer.

50. Schupp, J., T. Baugher, E. Winzeler, M. Schupp, and W. Messner, *Preliminary results with a vacuum assisted harvest system for apples.* Fruit Notes, 2011. **76**(4): p. 1-5.

51. Bergerman, M., S.M. Maeta, J. Zhang, G.M. Freitas, B. Hamner, S. Singh, and G. Kantor, *Robot Farmers: Autonomous Orchard Vehicles Help Tree Fruit Production.* IEEE Robotics & Automation Magazine, 2015. **22**(1): p. 54-63.

52. Ye, Y., L. He, and Q. Zhang, *A Robotic Platform "Bin-Dog" for Bin Management in Orchard Environment*, in *2016 ASABE Annual International Meeting*. 2016, ASABE: St. Joseph, MI. p. 1.

53. Wang, Z., L. Gong, Q. Chen, Y. Li, C. Liu, and Y. Huang. *Rapid Developing the Simulation and Control Systems for a Multifunctional Autonomous Agricultural Robot with ROS*. in *International Conference on Intelligent Robotics and Applications*. 2016. Springer.

54. Pitla, S.K., J.D. Luck, and S.A. Shearer, *Multi-Robot System Control Architecture (MRSCA) for Agricultural Production.* 2010 Pittsburgh, Pennsylvania, June 20-June 23, 2010, 2010: p. 1.

55. Gautam, A. and S. Mohan. *A review of research in multi-robot systems*. in *Industrial and Information Systems (ICIIS), 2012 7th IEEE International Conference on*. 2012.

56. Rodrigues, N., A. Pereira, and P. Leitão, *Adaptive Multi-Agent System for a Washing Machine Production Line*, in *Industrial Applications of Holonic and Multi-Agent Systems: 6th International Conference, HoloMAS 2013, Prague, Czech Republic, August 26-28, 2013. Proceedings*, V. Mařík, J.L.M. Lastra, and P. Skobelev, Editors. 2013, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 212-223.

57. Kanda, T., H. Ishiguro, T. Ono, M. Imai, and K. Mase. *Multi-robot cooperation for human-robot communication*. in *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*. 2002.

58. Holland, O. and C. Melhuish, *Stigmergy, Self-Organization, and Sorting in Collective Robotics.* Artificial Life, 1999. **5**(2): p. 173-202.

59. Langerwisch, M., T. Wittmann, S. Thamke, T. Remmersmann, A. Tiderko, and B. Wagner. *Heterogeneous teams of unmanned ground and aerial robots for reconnaissance and surveillance - A field experiment*. in *Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium on*. 2013.

60. Cao, Y.U., A.S. Fukunaga, and A. Kahng, *Cooperative Mobile Robotics: Antecedents and Directions.* Autonomous Robots, 1997. **4**(1): p. 7-27.

61. Garro, B.A., H. Sossa, and R.A. Vazquez. *Evolving ant colony system for optimizing path planning in mobile robots*. in *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007)*. 2007. IEEE.

62. Dias, M.B. and A. Stentz, *A market approach to multirobot coordination*. 2000, DTIC Document.

63. Noguchi, N., J. Will, J. Reid, and Q. Zhang, *Development of a master–slave robot system for farm operations.* Computers and Electronics in Agriculture, 2004. **44**(1): p. 1-19.

64. Fidan, B., C. Yu, and B. Anderson, *Acquiring and maintaining persistence of autonomous multi-vehicle formations.* IET Control Theory & Applications, 2007. **1**(2): p. 452-460.

65. Ma, M. and Y. Yang. *Adaptive triangular deployment algorithm for unattended mobile sensor networks*. in *International Conference on Distributed Computing in Sensor Systems*. 2005. Springer.

66. Cheng, L., Z.-G. Hou, and M. Tan. *Decentralized adaptive leader-follower control of multi-manipulator system with uncertain dynamics*. in *Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE*. 2008. IEEE.

67. Emmi, L., M. Gonzalez-de-Soto, G. Pajares, and P. Gonzalez-de-Santos, *New trends in robotics for agriculture: integration and assessment of a real fleet of robots.* The Scientific World Journal, 2014. **2014**.

68. Bailey, T., M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte. *Decentralised cooperative localisation for heterogeneous teams of mobile robots*. in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. 2011. IEEE.

69. Capodieci, N. and G. Cabri. *Collaboration in swarm robotics: A visual communication approach*. in *2013 International Conference on Collaboration Technologies and Systems (CTS)*. 2013.

70. He, L., R. Arikapudi, F.K. Anjom, and S.G. Vougioukas. *Worker position tracking for safe navigation of autonomous orchard vehicles using active ranging*. in *American Society of Agricultural and Biological Engineers Annual International Meeting 2014, ASABE 2014, July 13, 2014 - July 16, 2014*. 2014. Montreal, QC, Canada: American Society of Agricultural and Biological Engineers.

71. Acevedo, J.J., B.C. Arrue, I. Maza, and A. Ollero. *Cooperative perimeter surveillance with a team of mobile robots under communication constraints*. in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. 2013. IEEE.

72. Jensen, M.A.F., D. Bochtis, C.G. Sørensen, M.R. Blas, and K.L. Lykkegaard, *In-field and inter-field path planning for agricultural transport units.* Computers & Industrial Engineering, 2012. **63**(4): p. 1054-1061.

73. Stroupe, A., T. Huntsberger, A. Okon, H. Aghazarian, and M. Robinson. *Behavior-based multi-robot collaboration for autonomous construction tasks*. in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*. 2005.

74. Palm, R., A. Bouguerra, M. Abdullah, and A.J. Lilienthal. *Navigation in human-robot and robot-robot interaction using optimization methods*. in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2016.

75. Jorge, V.A.M., V.F. Rey, R. Maffei, S.R. Fiorini, J.L. Carbonera, F. Branchi, J.P. Meireles, G.S. Franco, F. Farina, T.S. da Silva, M. Kolberg, M. Abel, and E. Prestes, *Exploring the IEEE ontology for robotics and automation for heterogeneous agent interaction.* Robotics and Computer-Integrated Manufacturing, 2015. **33**: p. 12-20.

76. Hristoskova, A., C.E. Aguero, M. Veloso, and F. De Turck, *Heterogeneous context-aware robots providing a personalized building tour.* International Journal of Advanced Robotic Systems, 2013. **10**.

77. Zealand/Rabobank, F.F.o.N., *Farm Employee Remuneration Survey*. 2014: New Zealnd.

78. NZ, B.G. *Fruit Picking Jobs in New Zealand: Wages Explained*. 2019 [cited 2019 03, Feb]; Available from: https://www.backpackerguide.nz/fruit-picking-jobs-in-new-zealand-wages-explained/.

79. Tractor, C. *Tractors and Machinery*. 2016 [cited 2019 3/02]; Available from: http://www.capitaltractors.co.nz/.

80. Edwards, W.M., *Machinery management: estimating farm machinery costs*. 2001, Iowa State University, Department of Economics.

81. Afsharnia, F., M.A. Asoodar, A. Abdeshahi, and A. Marzban, *Failure rate analysis of four agricultural tractor models in southern Iran.* Agricultural Engineering International: CIGR Journal, 2013. **15**(4): p. 160-170.

82. Dijkstra, E.W., *A note on two problems in connexion with graphs.* Numerische mathematik, 1959. **1**(1): p. 269-271.

83. White, K. *Why are so few Brits prepared to pick fruit?* 2017 [cited 2017 30 Jun]; Available from: https://www.thegrocer.co.uk/people/brexit-and-the-workforce/why-are-so-few-brits-prepared-to-pick-fruit/554452.article.

84. Cormen, T.H., C.E. Leiserson, R.L. Rivest, and C. Stein, *Dijkstra's algorithm*, in *Introduction to Algorithms 2nd edition*. MIT Press. p. 595-599.

85. Hagberg, A., D. Schult, and P. Swart, *Exploring network structure, dynamics, and function.* 2009.

86. Trémaux, C.P. *École Polytechnique of Paris (X: 1876)*. in *Proc. French Engineer of the Telegraph in Public Conference*. 2010.

87. McKenzie, R., *Agricultural soil compaction: Causes and management.* Alberta Agriculture and Forestry, 2010.

88. Ball, D., P. Ross, A. English, T. Patten, B. Upcroft, R. Fitch, S. Sukkarieh, G. Wyeth, and P. Corke. *Robotics for sustainable broad-acre agriculture*. in *Field and Service Robotics*. 2015. Springer.

89. Grimstad, L., C.D. Pham, H.N.T. Phan, and P.J. From. *On the design of a low-cost, light-weight, and highly versatile agricultural robot*. in *ARSO*. 2015.

90. Sundarapandian, V., *Probability, statistics and queuing theory*. 2009: PHI Learning Pvt. Ltd.

91. Erlang, A.K., *The theory of probabilities and telephone conversations.* Nyt. Tidsskr. Mat. Ser. B, 1909. **20**: p. 33-39.

92. Sztrik, J., *Basic queueing theory.* University of Debrecen, Faculty of Informatics, 2012. **193**.

93. Adan, I. and J. Resing, *Queueing theory*. 2002, Eindhoven University of Technology Eindhoven.

94. Cook, J.D. *Server utilization: Joel on queuing*. 2009 [cited 2019 25/02]; Available from: https://www.johndcook.com/blog/2009/01/30/server-utilization-joel-on-queuing/.

95. Hutching, G. *Tractor sales crash, price rises expected*. 2015 [cited 2015 27/07]; Available from: https://www.stuff.co.nz/business/farming/agribusiness/70581366/null.

96. COMPANIES, A. *Do You Know Your Average Forklift Cost per Hour?* [cited 2019 26/02]; Available from: https://atlastoyota.com/blog-posts/do-you-know-your-average-forklift-cost-per-hour/.

97. *The Optimal Time to Replace your Forklift*. 2012 [cited 2012 24/10]; Available from: https://www.aalhysterforklifts.com.au/index.php/about/blog-post/the_optimal_time_to_replace_your_forklift.

98. *General Rules about Forklift Pricing*. [cited 2013 23/02]; Available from: https://www.aalhysterforklifts.com.au/index.php/about/blog-post/general_rules_about_forklift_pricing.

99. *Electric quad bike for adults*. [cited 2019; Available from: https://www.alibaba.com/product-detail/2018-cheap-four-wheelers-electric-quad_60580257462.html?spm=a2700.7724857.normalList.6.679338aaViN0xX&s=p.

100. Localmatters. *Electric ATV takes on farming*. 2017; Available from: https://localmatters.co.nz/news/13861-electric-atv-takes-on-farming.html.

101. Andersen, J.C., *Mobile robot navigation*. 2007, Technical University of Denmark.

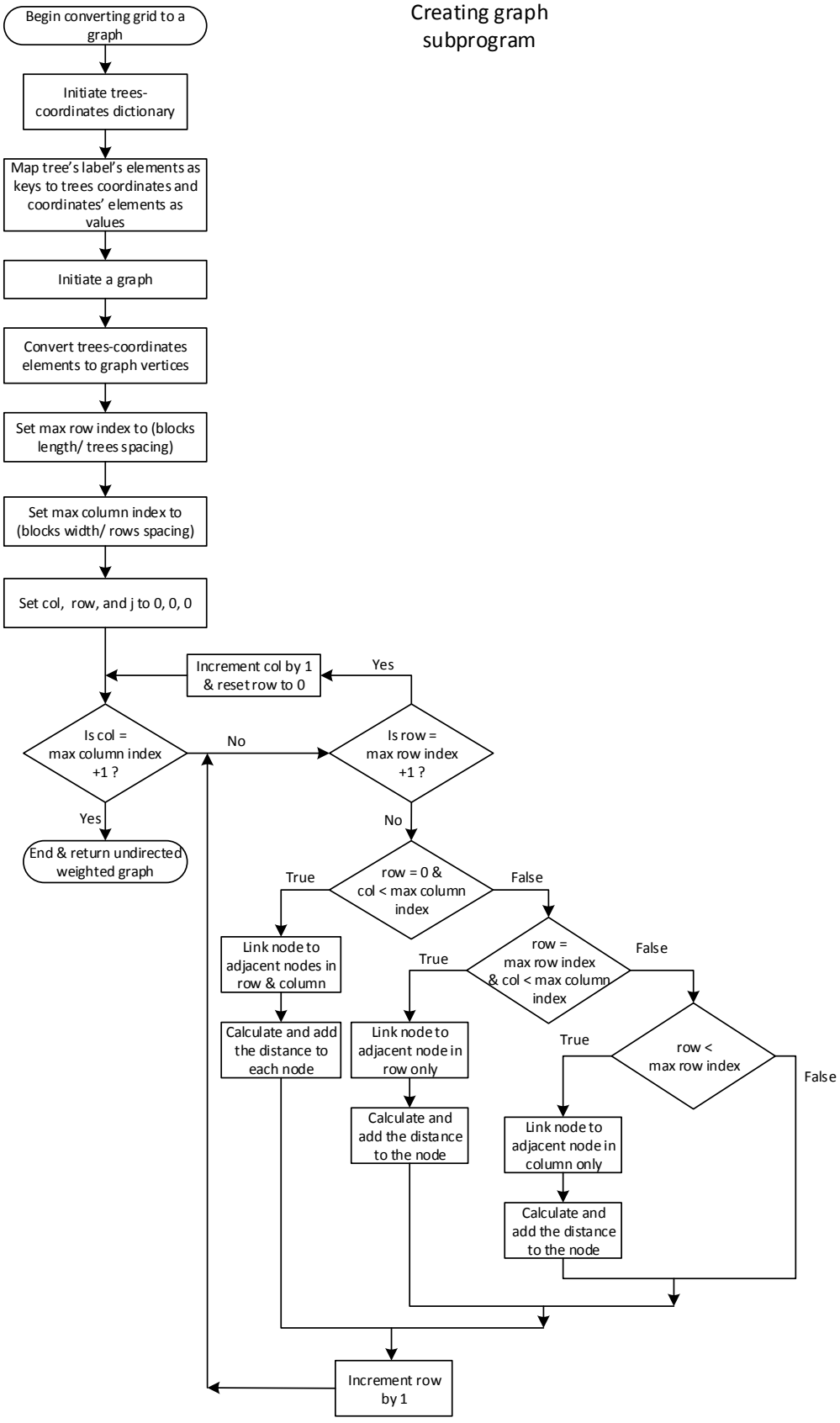
102. Dalal, N. and B. Triggs. *Histograms of oriented gradients for human detection*. in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. 2005. IEEE.

103. Errami, M. and M. Rziza. *Improving pedestrian detection using support vector regression*. in *2016 13th International Conference on Computer Graphics, Imaging and Visualization (CGiV)*. 2016. IEEE.

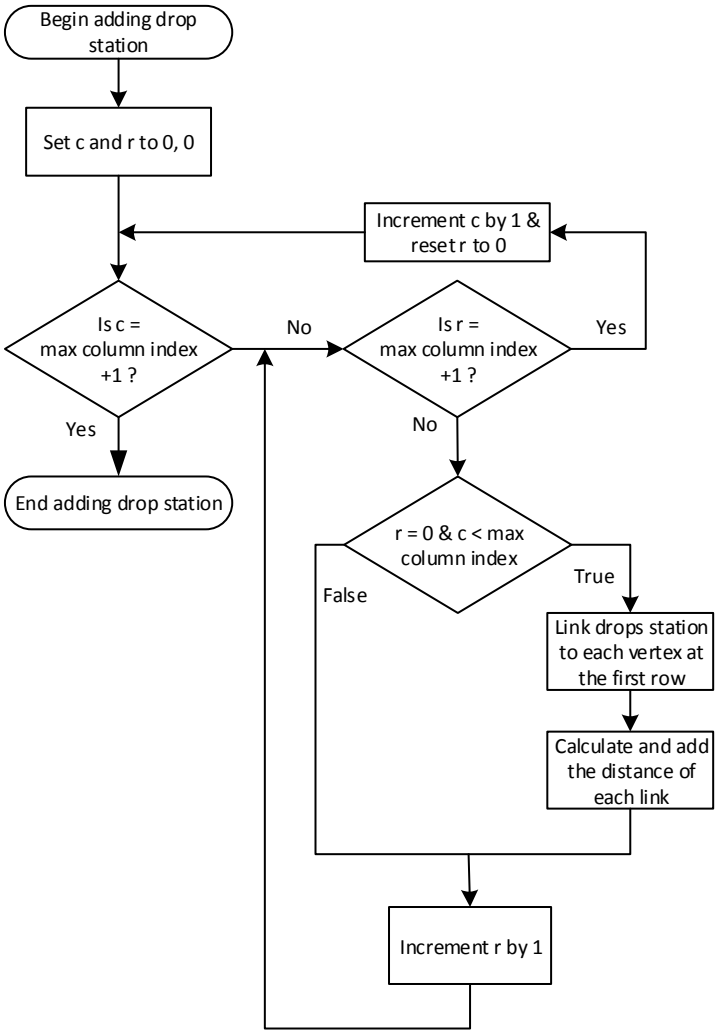
104. Stereolabs. *Depth Sensing - Advanced Settings*. 2018 [cited 2018 2018-02-14]; Available from: https://docs.stereolabs.com/overview/depth-sensing/advanced-settings/.

APPENDICES

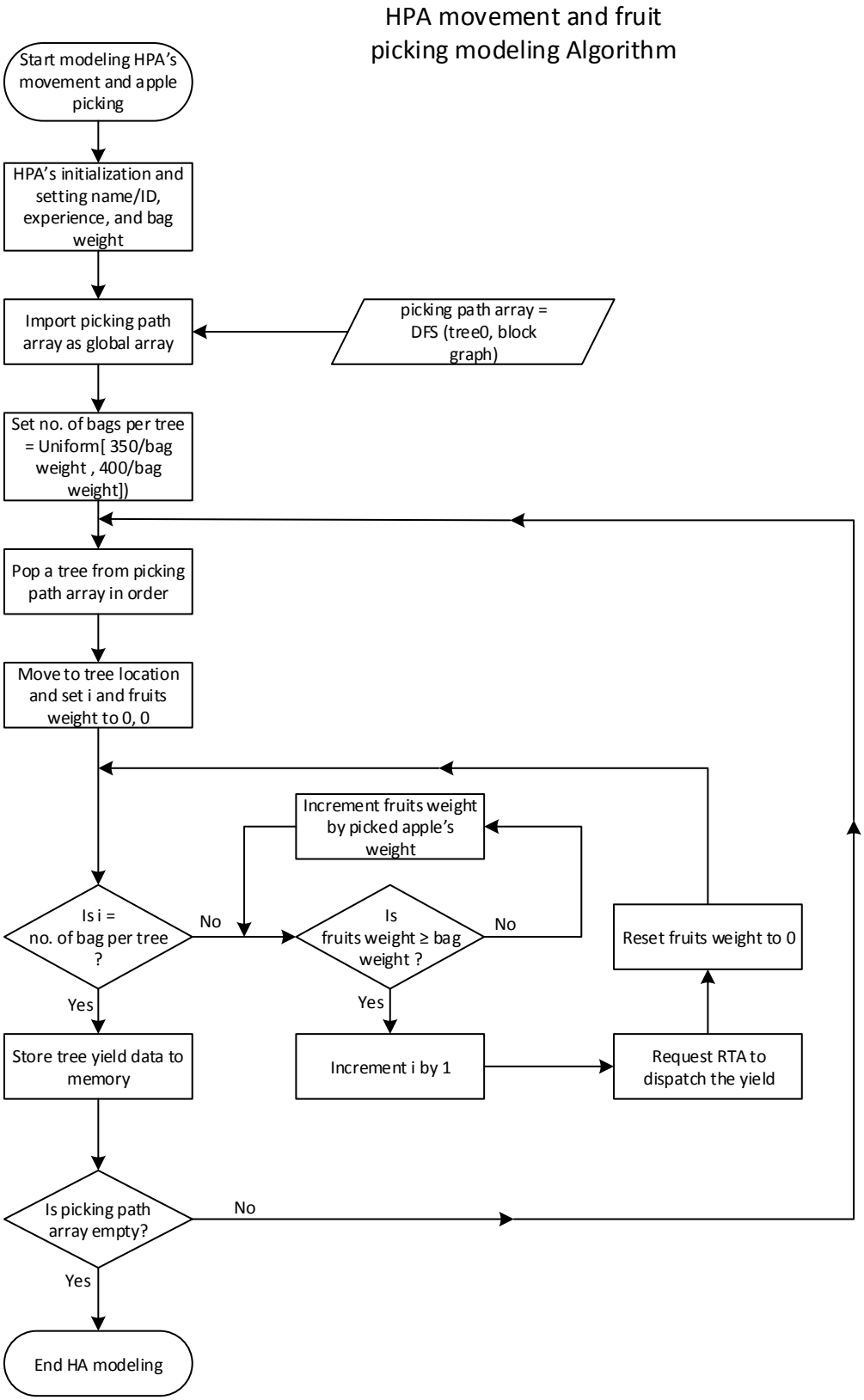
Appendix A1: Orchard block modelling algorithm flowchart



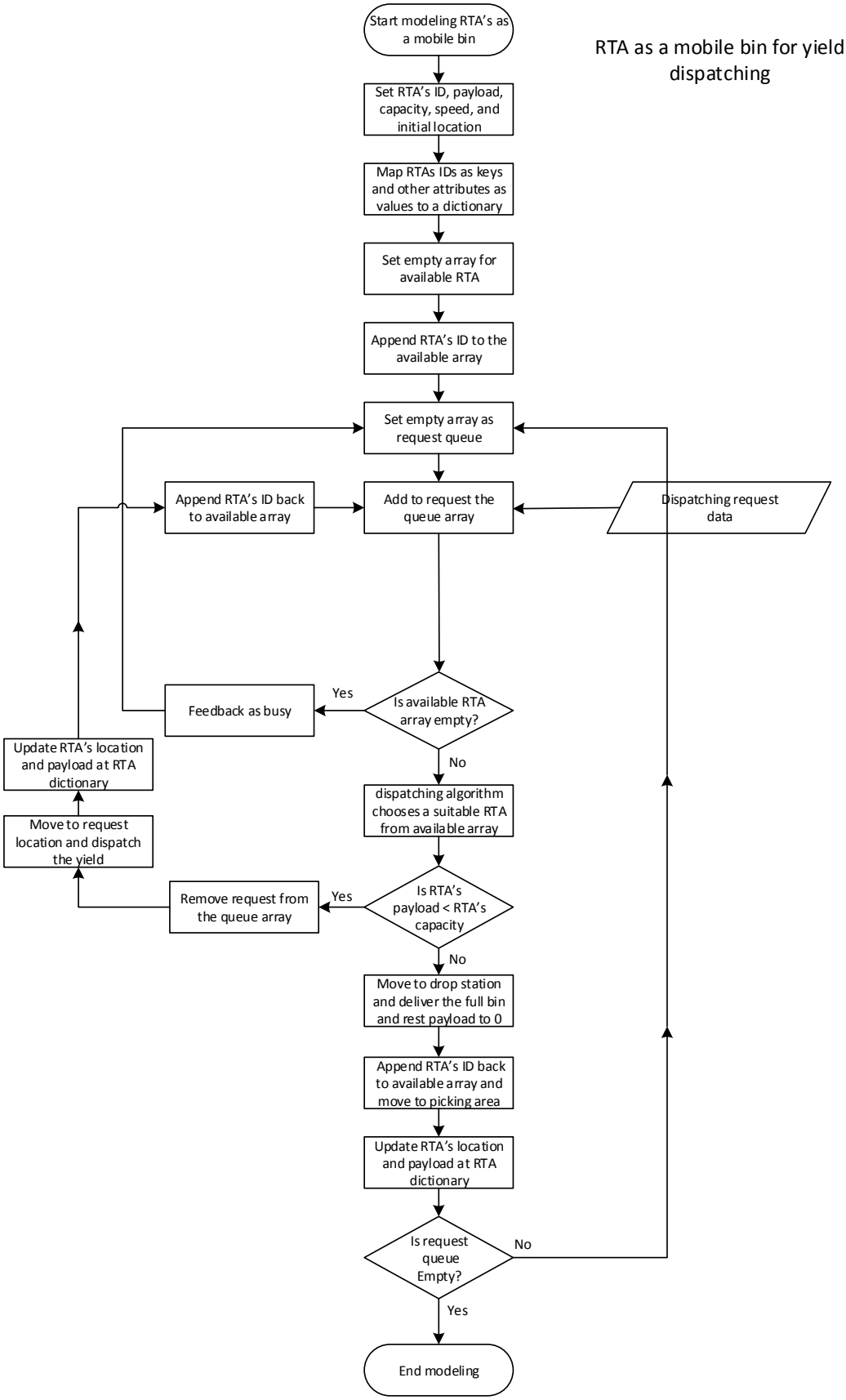
Adding drop station subprogram



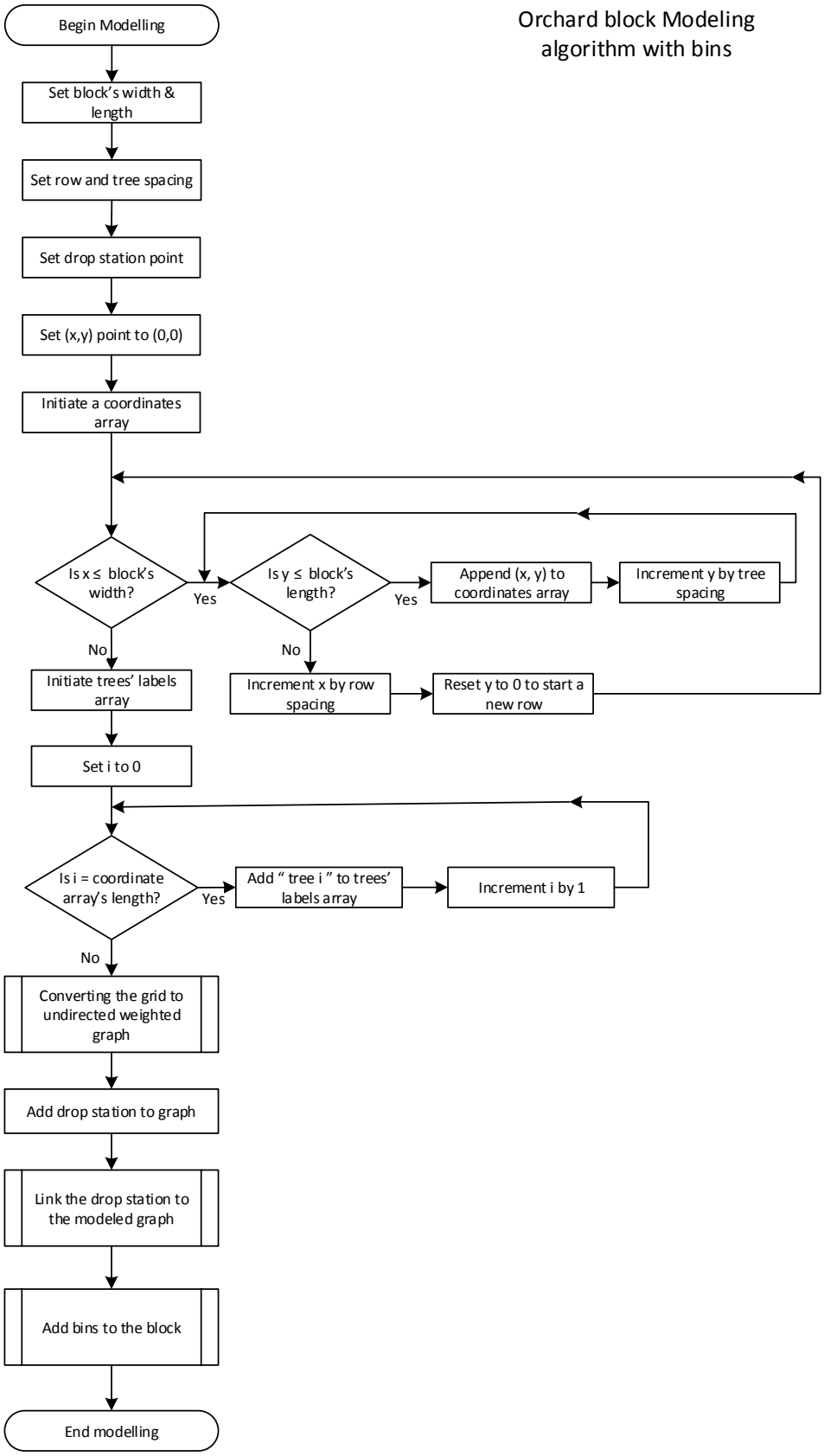
Appendix A2: HPA's modelling algorithm



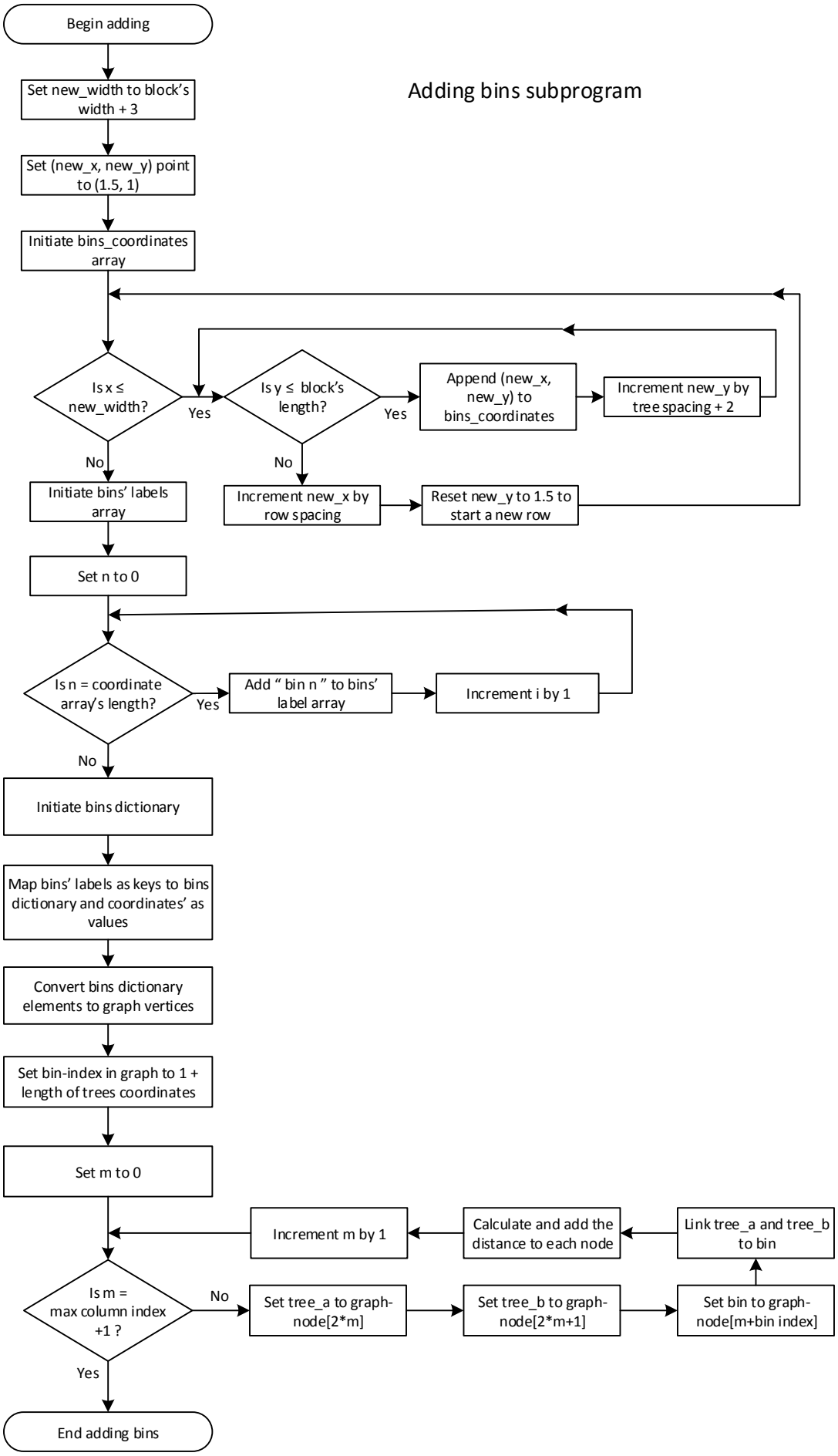
Appendix A3: RTA's modelling algorithm



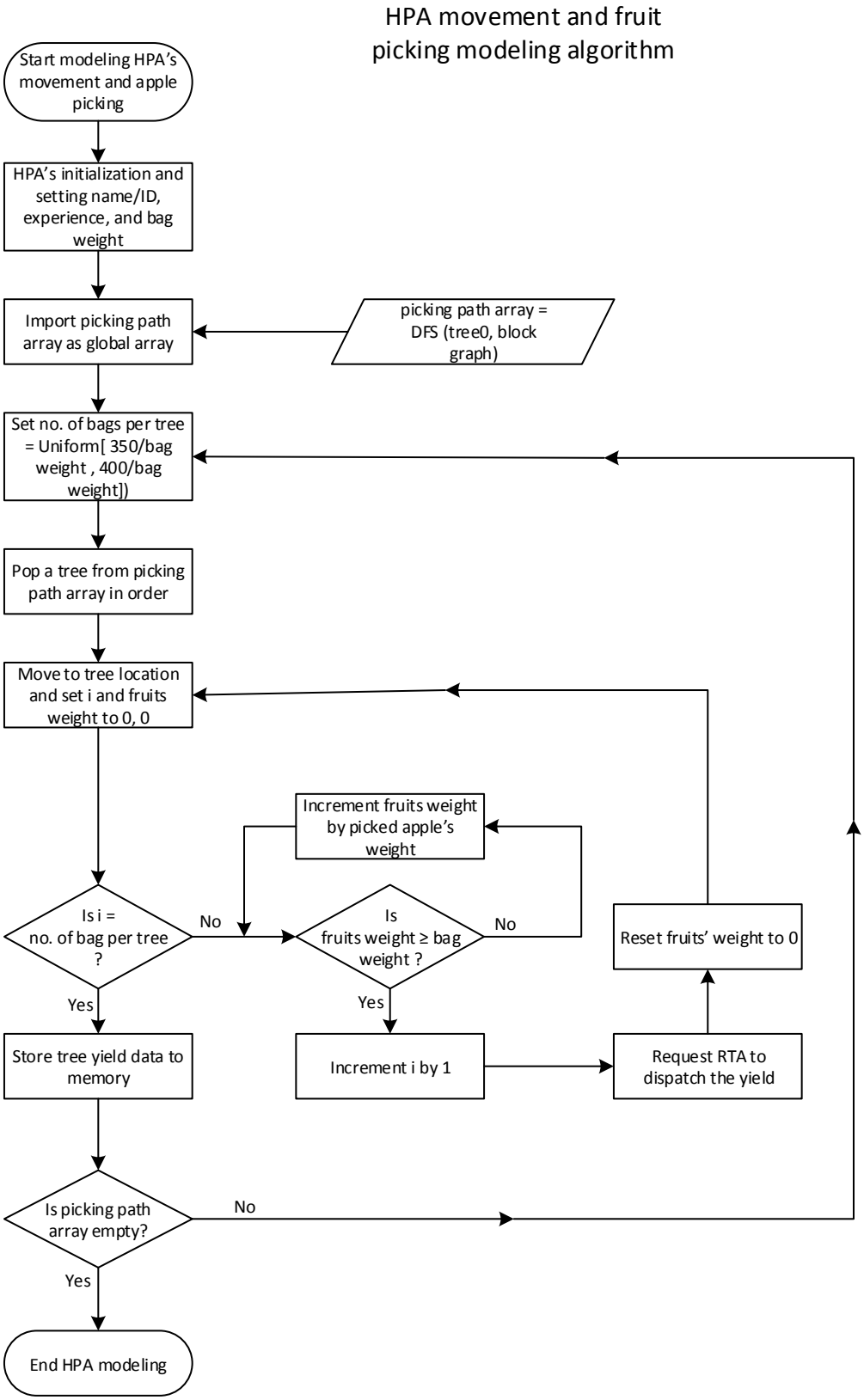
Appendix A4: Orchard block modelling algorithm with bins flowchart



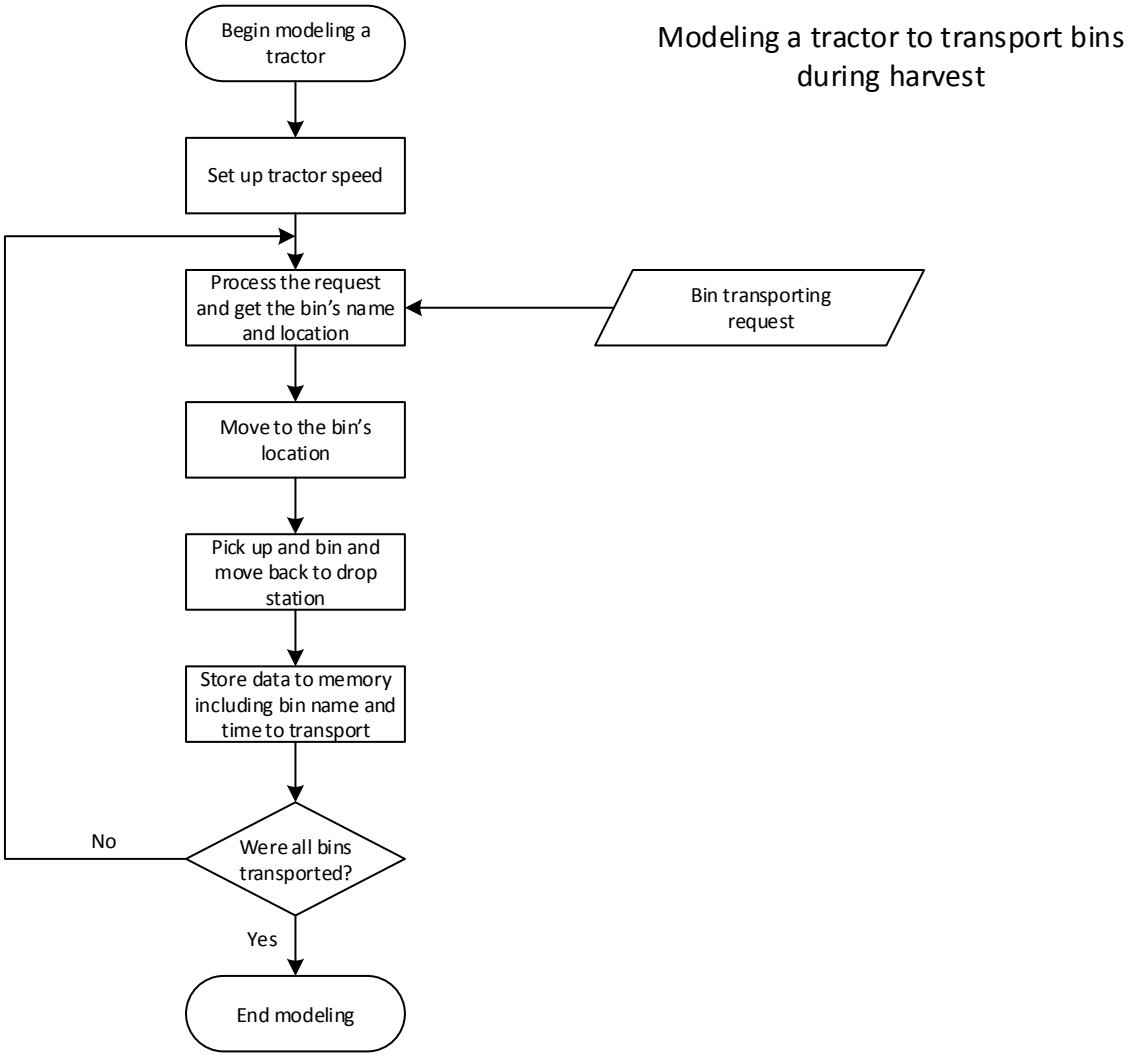
Adding bins subprogram



Appendix A5: Modified HPA's modelling algorithm for tractor yield management



Appendix A6: Tractor modelling algorithm



Appendix B1: Site visit to Plant & Food Company's orchard on 272 Whakarewa St Motueka, 7196, New Zealand.

The visit to the plant food and research orchard in Motueka Issues and Motivations

Motivation to implement multi-agent transportation system

- The predictable human-agent movement pattern during harvesting
- The easy access to the orchard trees' row form both ends
- The amount of produced apples and the transportation methods can be automated
- The amount of information a robotic transporting agent can share during each task regarding, harvested trees, blocks, and the whole orchard
- The framework and the robot can be used for different tasks during the non-harvesting season such as spraying, fertilizing...etc.

The issues highlighted by the orchard manager

- If a system is running with an autonomous vehicle, a good ground must be prepared with good drainage and solid flat surfaces.
- Buttholes, trenches, and muddy wet ways make tractors stuck and they need to be towed out. A tractor with a human driver will assist and manage to get the tractor out but they still get stuck sometimes.
- Mapping and sensors placing on the orchard.
- The supervisor or quality control should be the one to call the robot to pick up the bin. Since workers might call to transport the bin while it is not full.

Other Issues that were not mentioned by the manager if using a multi-agent system

Transportation:

- The robotic transporting agent should transport empty bins to the tree rows and transport back full bins.
- The robotic transporting agent should have the ability to identify a full bin.
- The robotic transporting agent would only move along the rows and not across the rows since rows have trees' supporting poles and wires.
- The robotic transporting agent should avoid or handle ground obstacles.

Navigation and communication

- The robotic transporting agent will be equipped with GPS and have wireless communication coverage.
- There is a chance that the surrounding trees, trees' supporting poles, and supporting wires would block the GPS signal.

Size of the robot and its load capacity:

- The size of the robotic transporting agent should consider the width of the tree rows and the clearance spaces since during the visit it was observed two different trees arrangements with different spaces and canopy
- The loading capacity of the robot could handle whether to use large robots or smaller ones.

Positioning the unit load when transporting:

- The bins should be positioned on the robot during transporting in a way that It will not lose balance such as if the robot had forklifts, the robot weight should be higher than the bin. Heavier robots will maintain better stability during transport.

The interview summary

Mobility and Harvesting Movement Patterns:

1. The pickers move systematically in a standard pattern from end to end within each row in a block starting at the ripest block and tree by tree.
2. The average time of the whole orchard to be harvested is variable. An average size orchard takes 21-28 days. The orchard is divided into blocks. Each block takes 3-4 days to be completely harvested by pickers
3. Most orchard has a single collection station. There will be another collection station if the harvested blocks are far more than 750m.

Fruit picking:

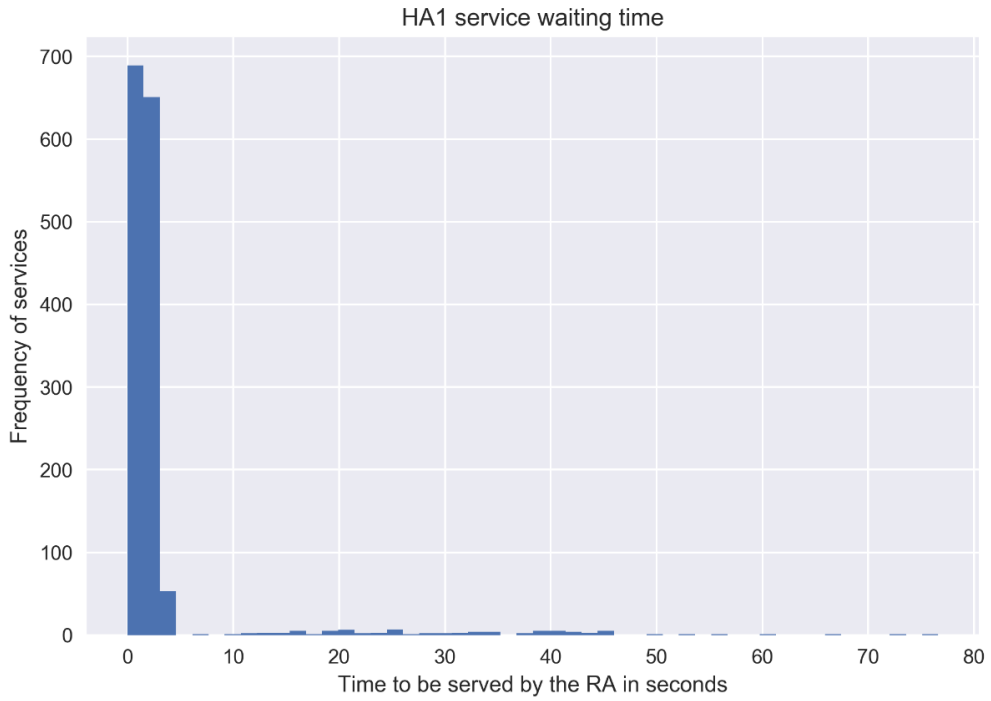
1. An apple orchard is harvested once a year.
2. An apple tree will be visited once every harvesting task. During the whole harvesting seasons, it will be visited a maximum of 4 times.
3. Visiting the tree more than 4 times is impractical as it causes trees and fruits damage.
4. The size of one hectare needs less than a day to be harvested.
5. 2 pickers will cover a hectare moving systematically a tree by three.
6. Each worker will visit a 100 tree per day picking 300 apples per tree.
7. A standard picker will collect 2 tons of apples each day but a good picker would collect 4 tons.
8. The pickers carry a bag to collect the apples and then empty to a wooden bin which is placed every 10-20m depending on the plantation intensity. While a tractor will move the bin when required to a closer distance to the pickers.
9. The wooden bin size is 1.2m length x 1.2m width x 800m height referred to C.A bins which mean controlled atmosphere bins. The empty bin weight 50-60kg.
10. It takes 1 hour to 90 minutes to fill up a bin and it depends also on how ripe the fruits are.

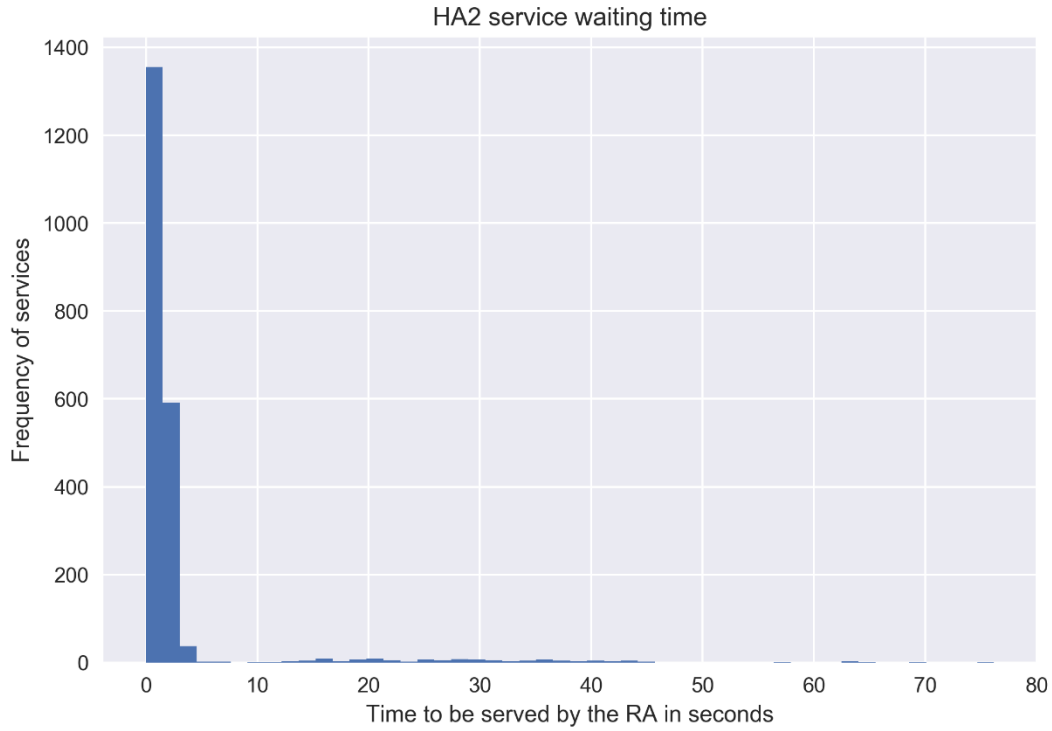
11. The maximum weight of a full bin is 300kgs to 350kgs with the number of 2500 apples.
12. A fully grown apple tree with a height of 3 meters will produce 300kg to 400 kg.
13. A typical commercial orchard has the size of 30 – 50 hectares.

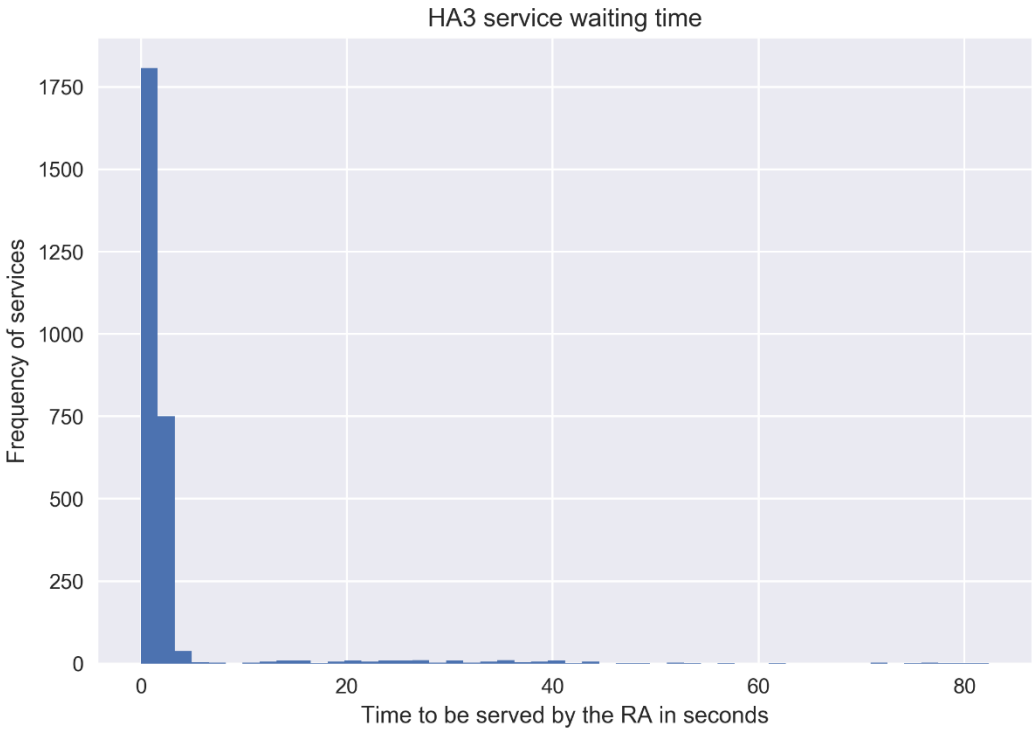
Fruit transportation:

1. Tractors are used to transport bins to the collection station. Standard tractors are used with forklifts on the front and the back to transport 2 bins at a time. Some orchard would have bigger tractors to carry 6 bins at a time.
2. On average it takes 10 mins to transport the bins to the station.
3. The average travel distance is 750m maximum.

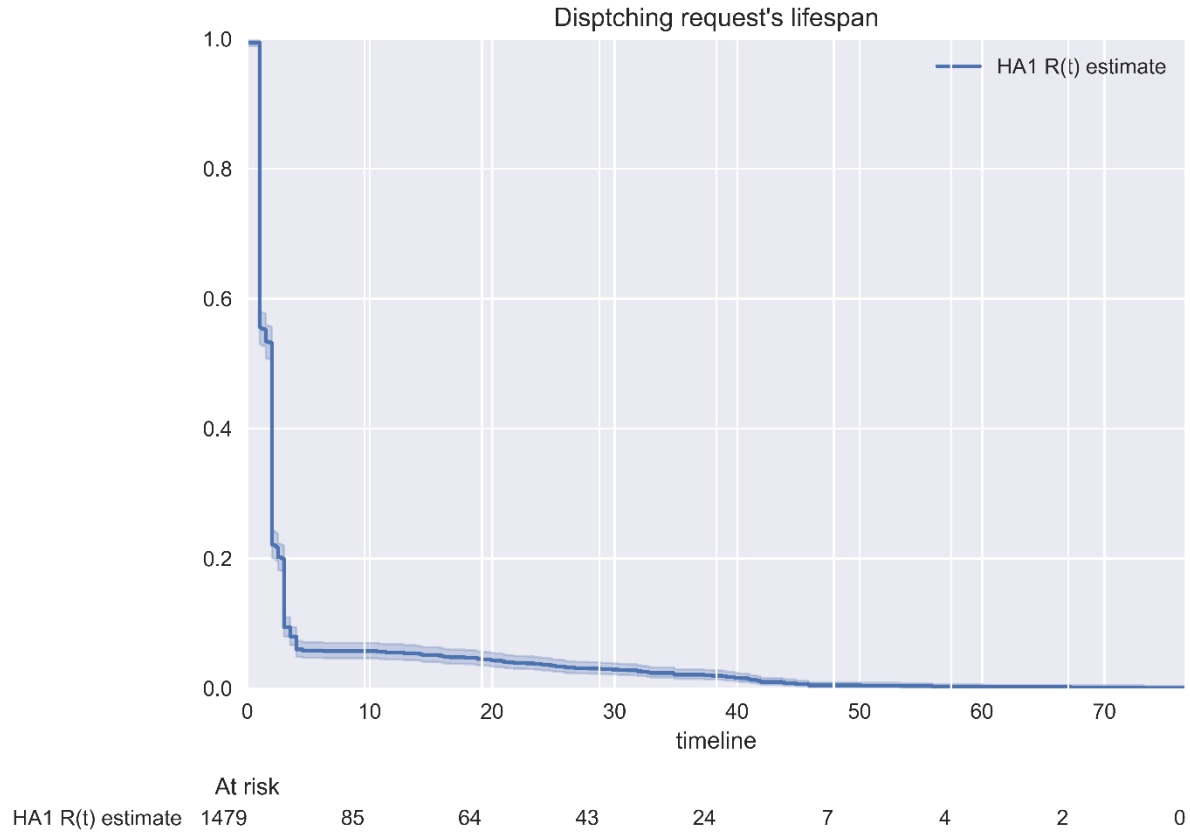
Appendix C1: The single RTA model scenario HPA1, HPA2, and HPA3 service waiting time distribution.

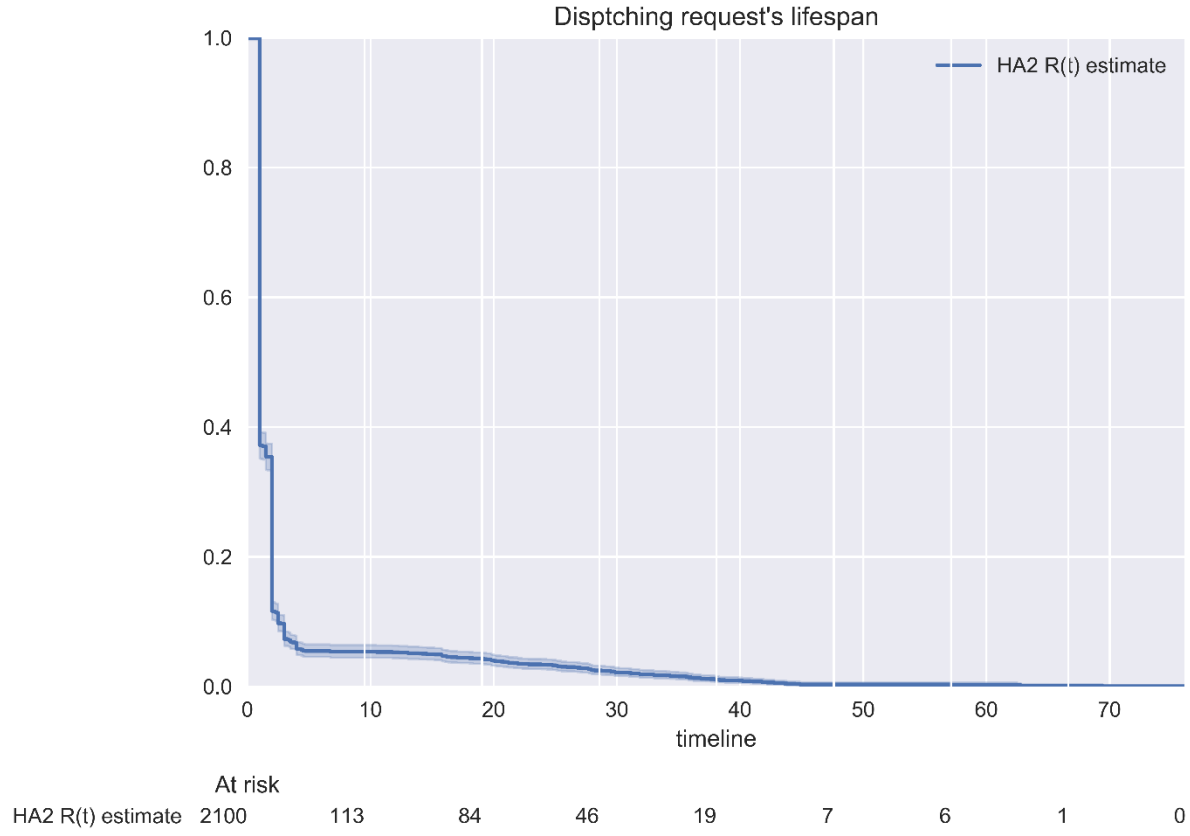


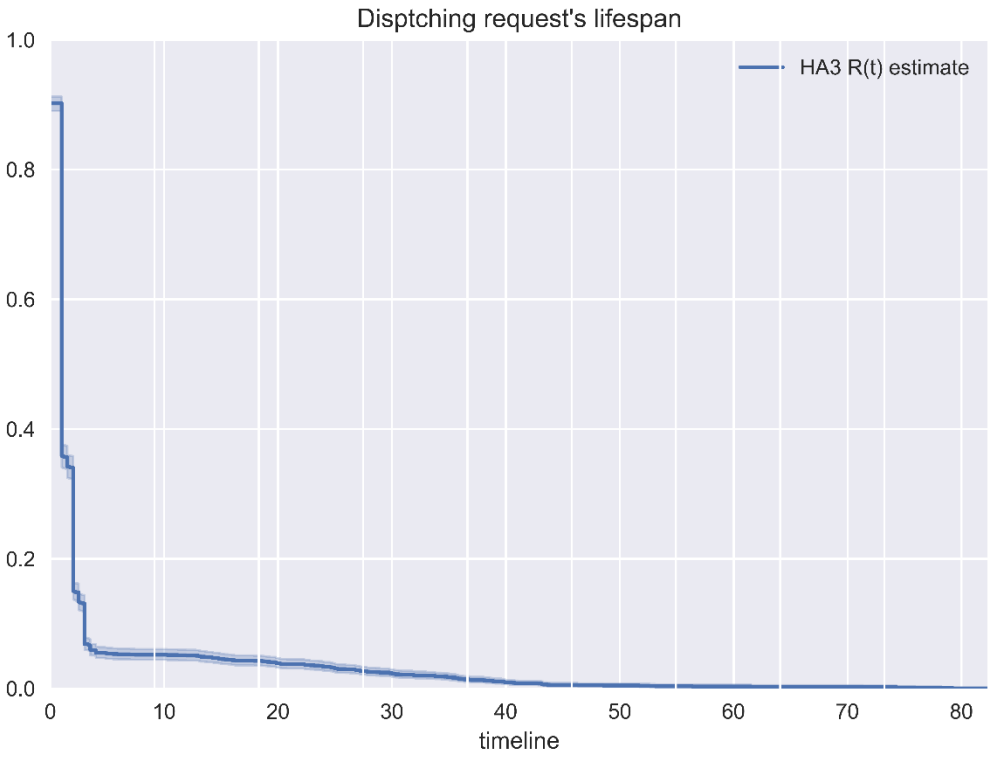



HA3 service waiting time
Frequency of services
1750
1500
1250
1000
750
500
250
0
0
20
40
60
80
Time to be served by the RA in seconds

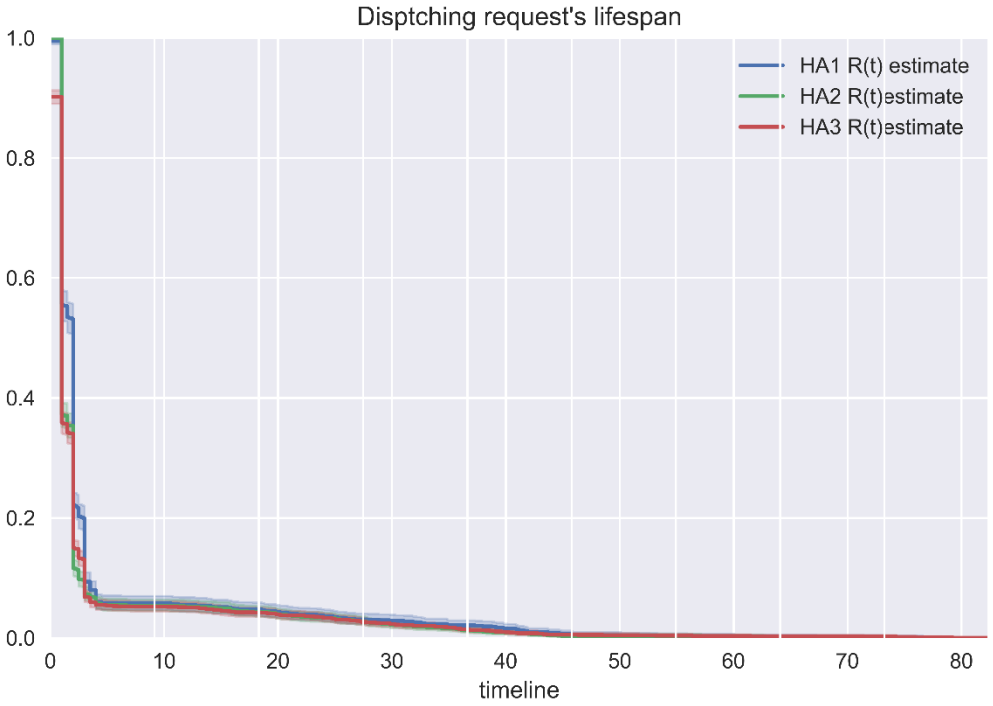
Appendix C2: The single RTA model reliability analysis employing.





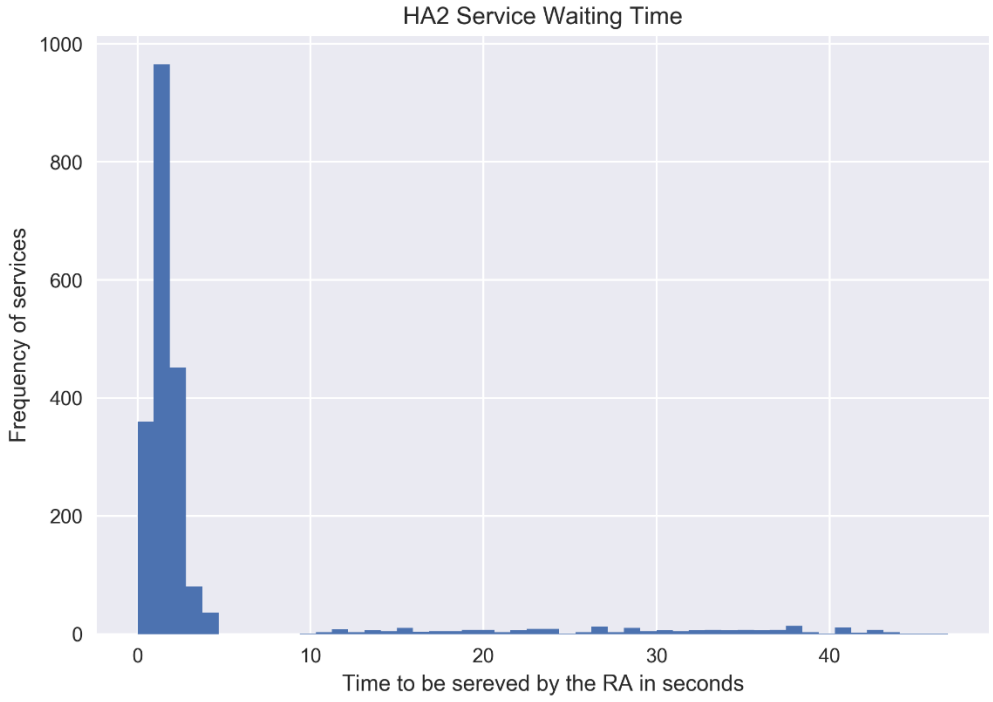


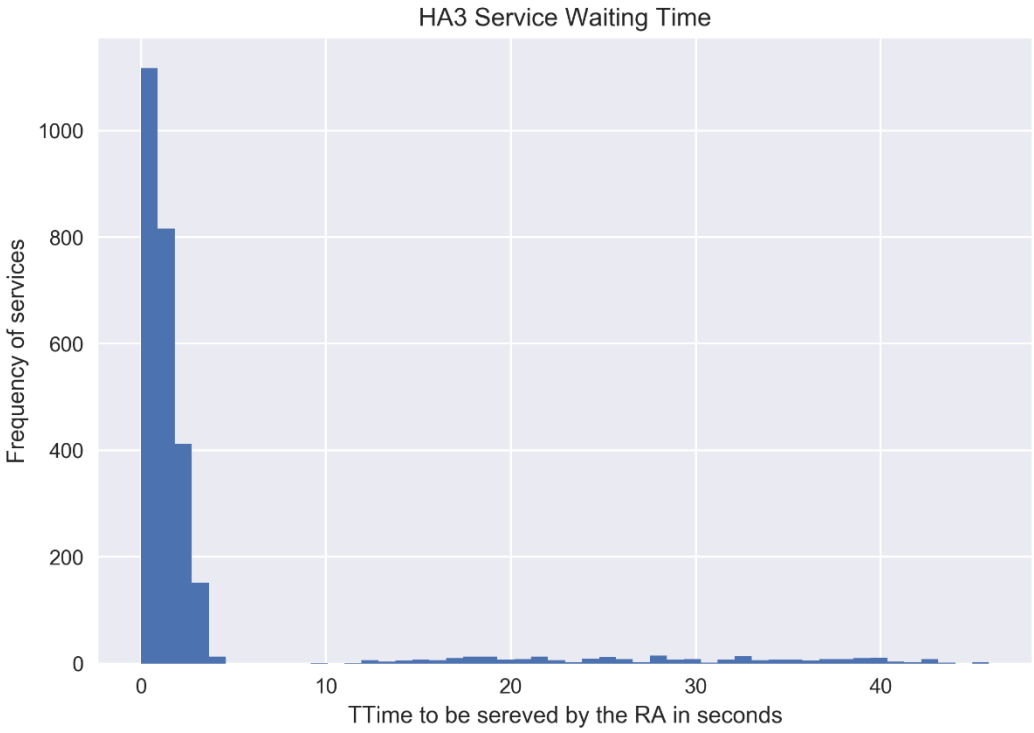
At risk										
HA3 R(t) estimate	2745	143	108	65	26	13	9	8	1	0

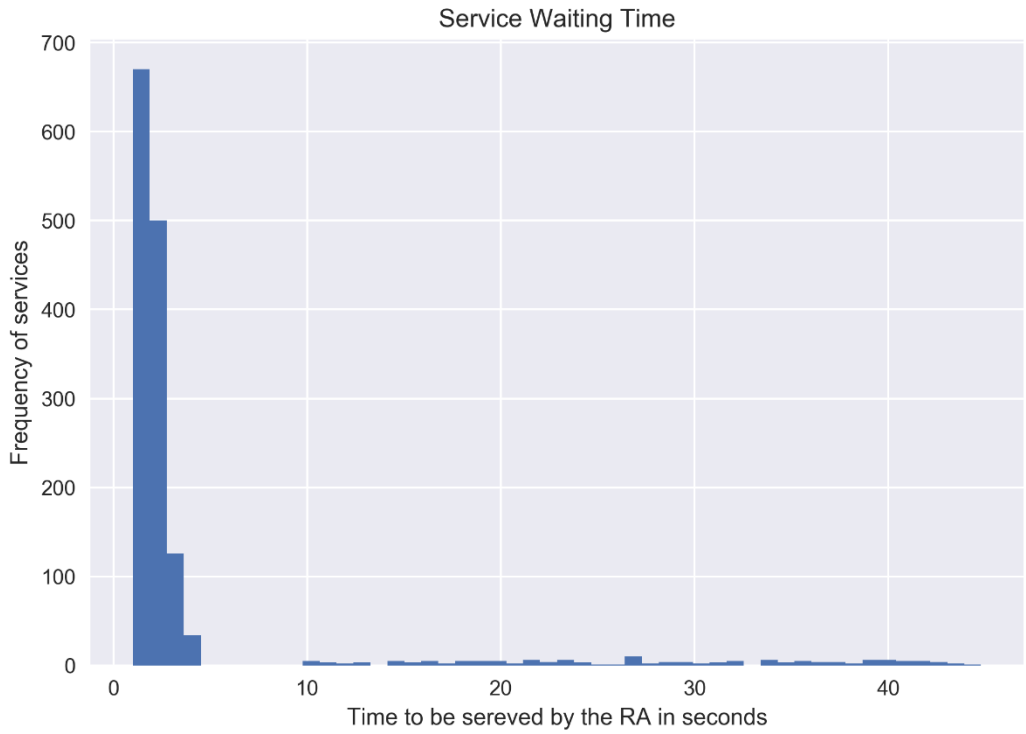


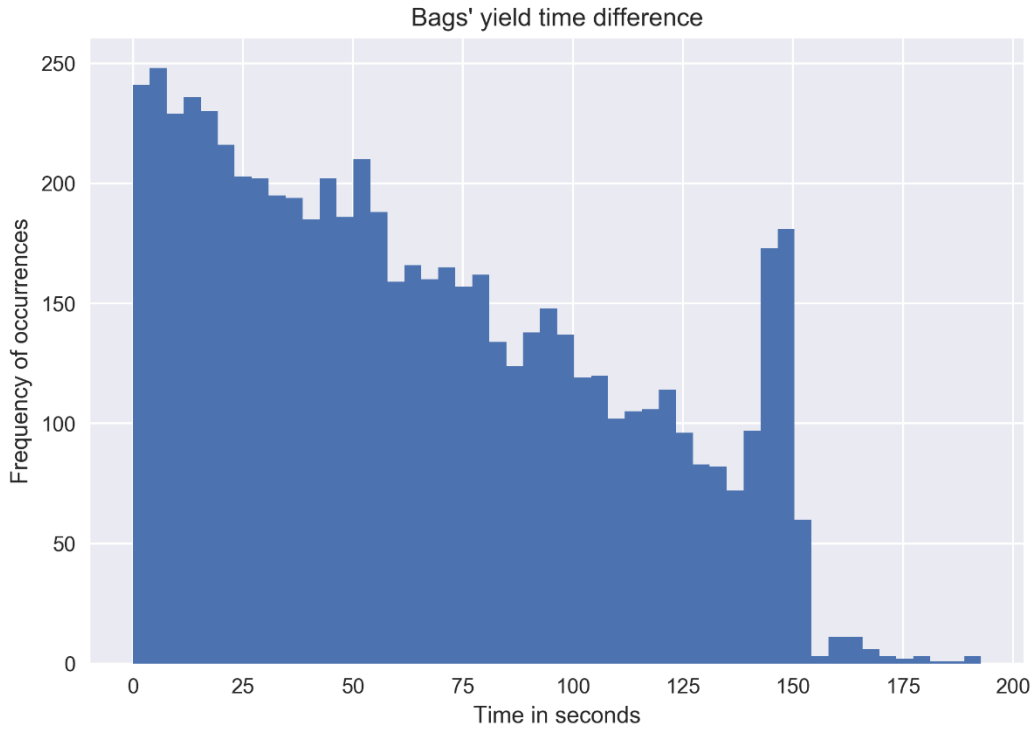
At risk										
HA1 R(t) estimate	1479	85	64	43	24	7	4	2	0	0
HA2 R(t) estimate	2100	113	84	46	19	7	6	1	0	0
HA3 R(t) estimate	2745	143	108	65	26	13	9	8	1	0

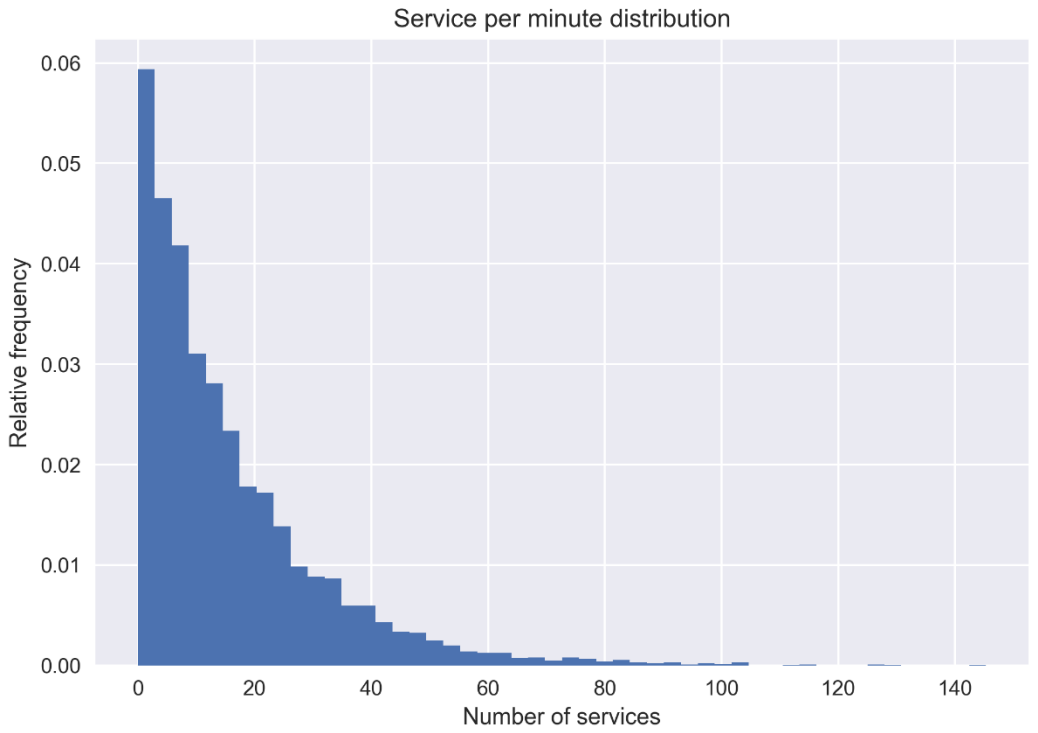
Appendix C3: Two RTA models scenario, service waiting time distribution for HPA1, HPA2, HPA3 and all of the HPAs waiting time combined.



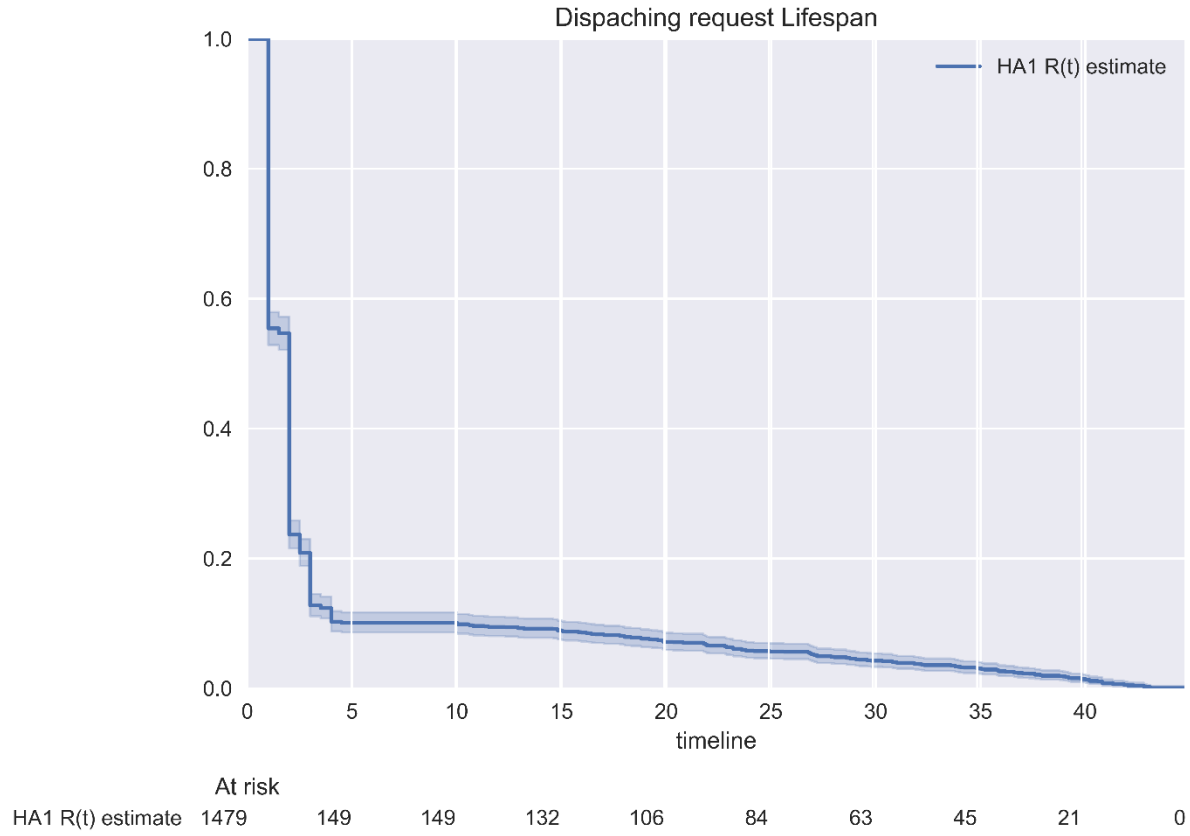

HA3 Service Waiting Time
Frequency of services
1000
800
600
400
200
0
0
10
20
30
40
TTime to be sereved by the RA in seconds

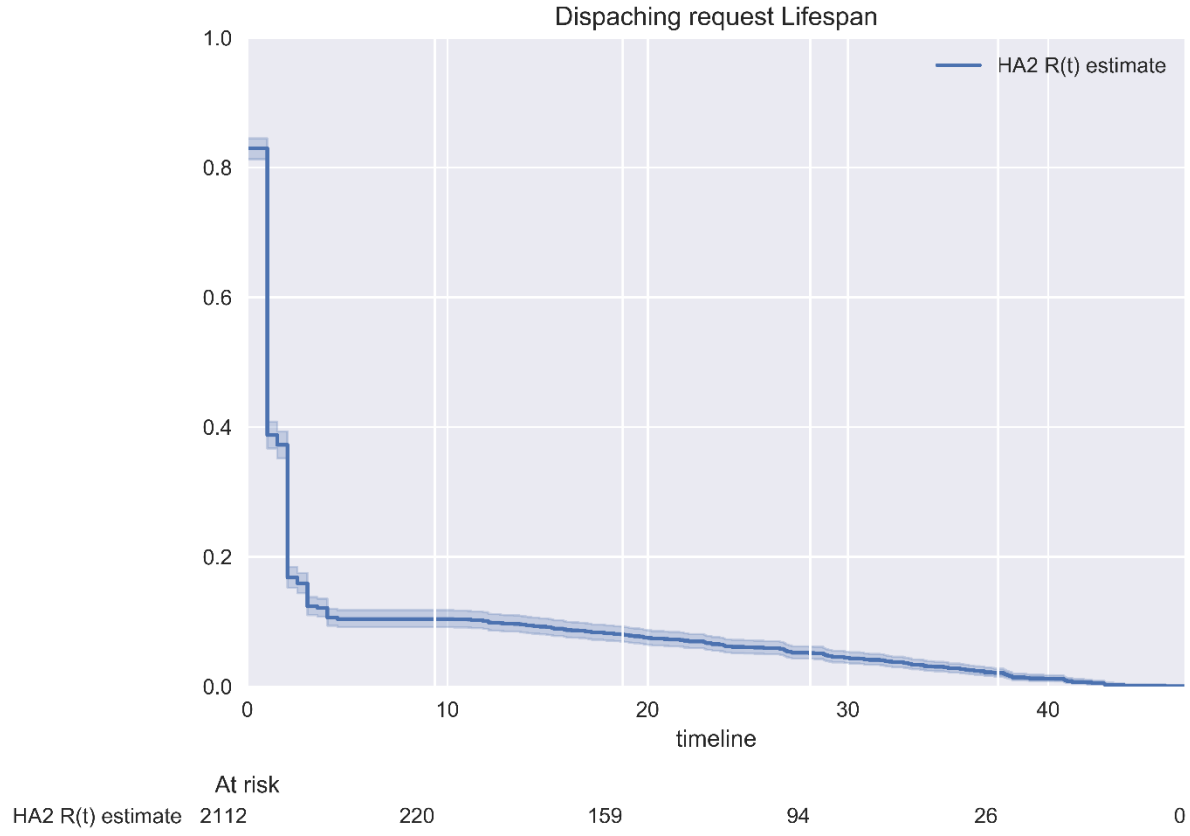

Service Waiting Time
Frequency of services
700
600
500
400
300
200
100
0
0
10
20
30
40
Time to be sereved by the RA in seconds

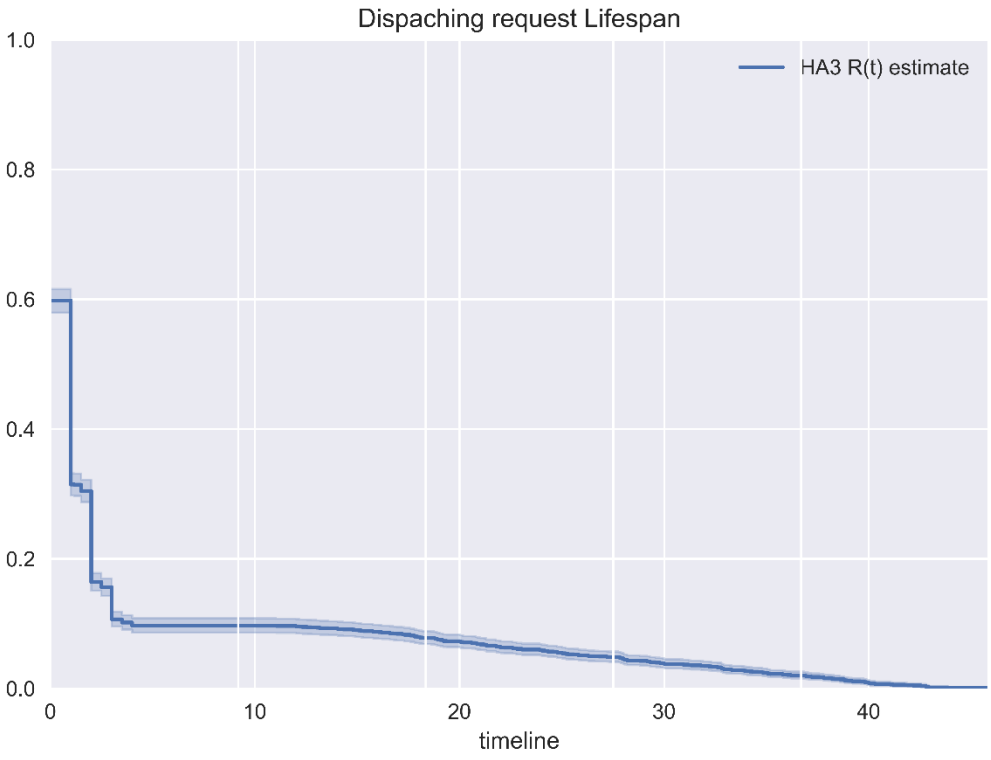

Bags' yield time difference
Frequency of occurrences
250
200
150
100
50
0
0
25
50
75
100
125
150
175
200
Time in seconds


Service per minute distribution
Relative frequency
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
20
40
60
80
100
120
140
Number of services

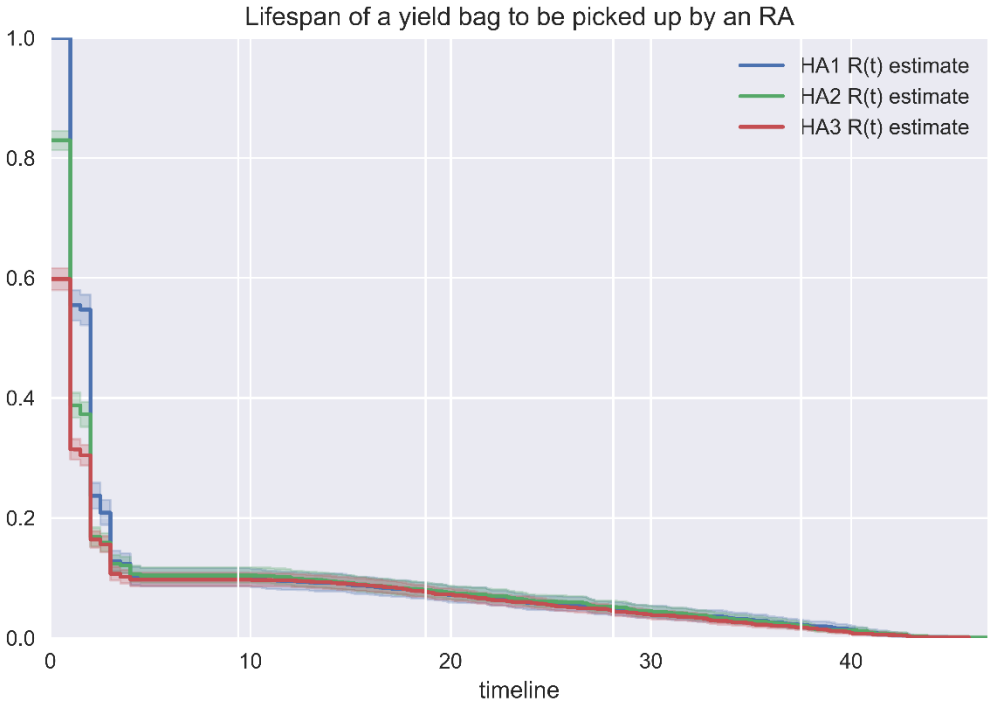
Appendix C4: The two RTA models reliability analysis.





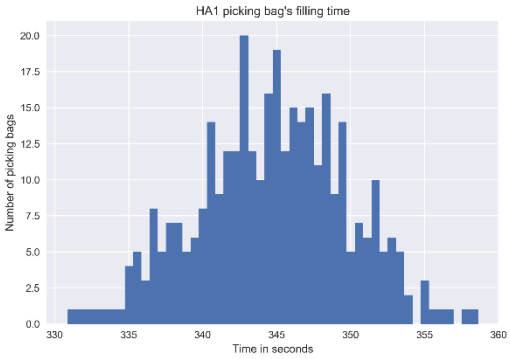


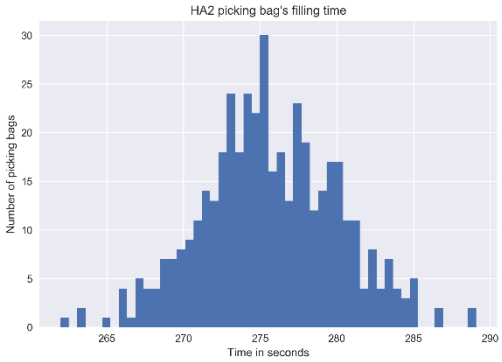
	At risk					
HA3 R(t) estimate	2779	270	203	108	25	0

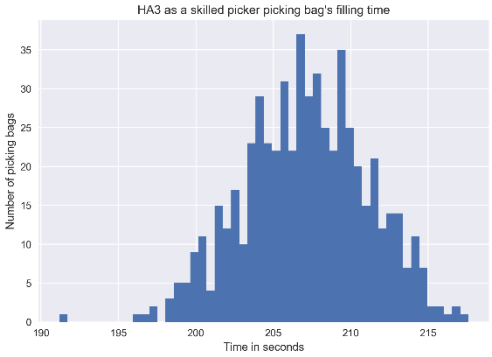


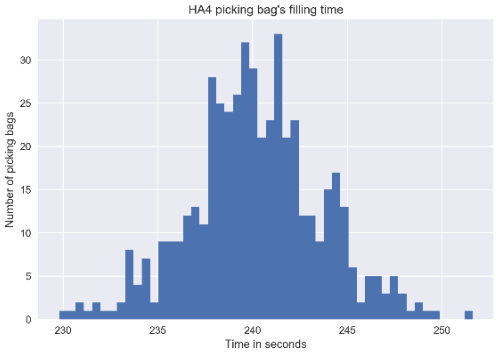
	At risk					
HA1 R(t) estimate	1479	149	106	63	21	0
HA2 R(t) estimate	2112	220	159	94	26	0
HA3 R(t) estimate	2779	270	203	108	25	0

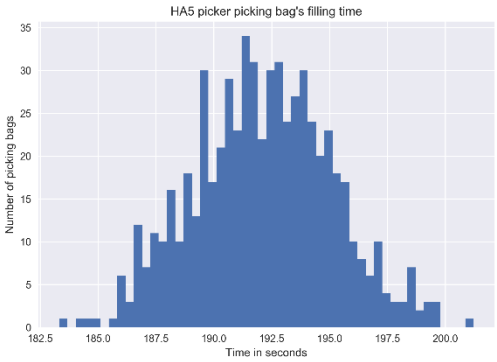
Appendix D1: FAFS algorithm simulation scenario HPA models request picking filling and service waiting time distribution.

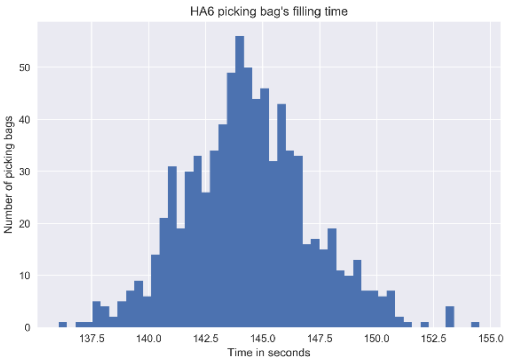


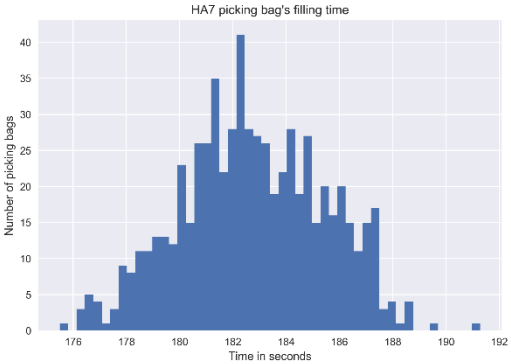


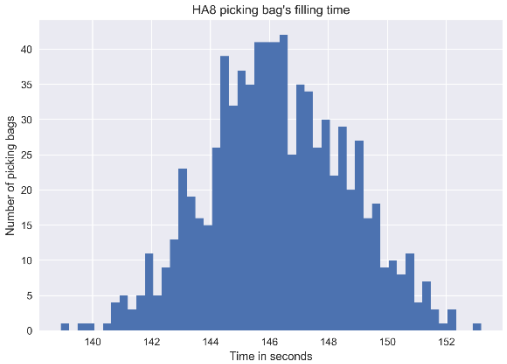


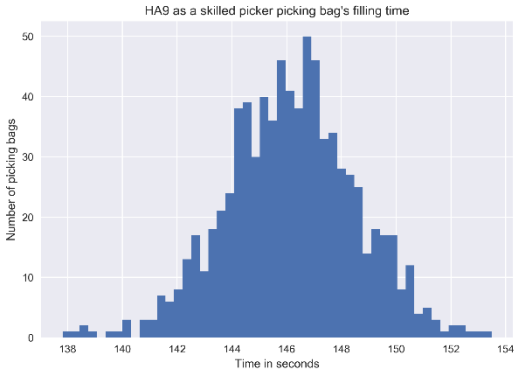


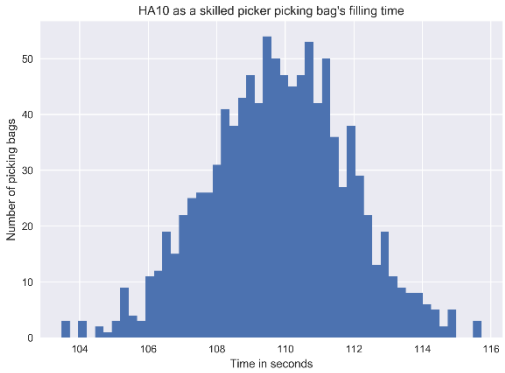


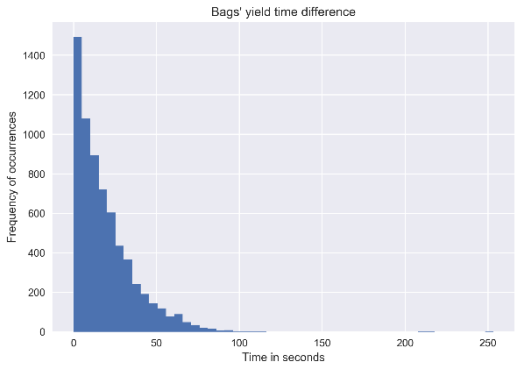


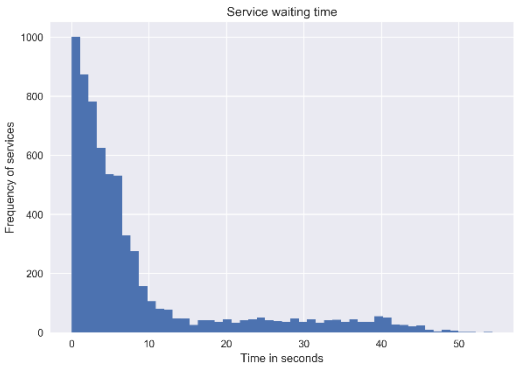


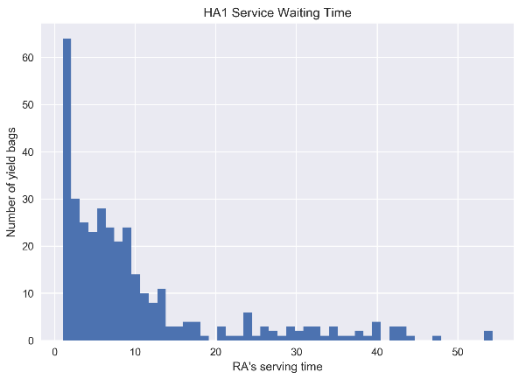


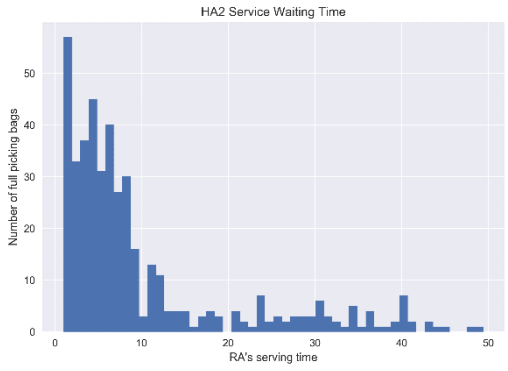

HA9 as a skilled picker picking bag's filling time
Number of picking bags
Time in seconds

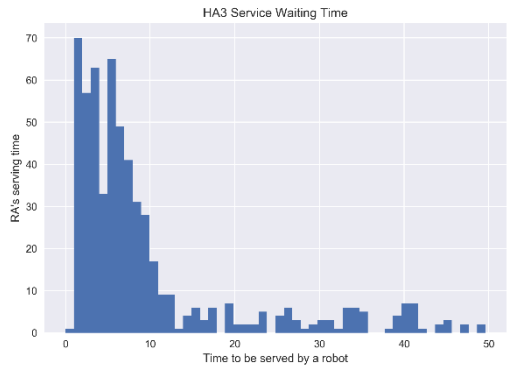

HA10 as a skilled picker picking bag's filling time
Number of picking bags
Time in seconds


Bags' yield time difference
Frequency of occurrences
Time in seconds

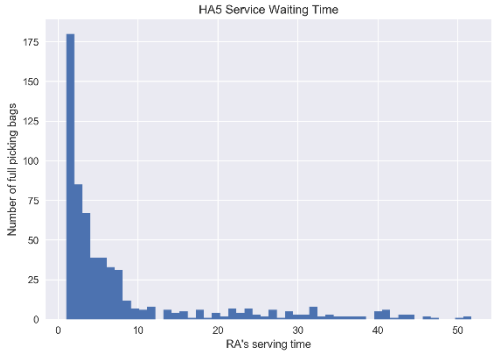

Service waiting time
Frequency of services
Time in seconds

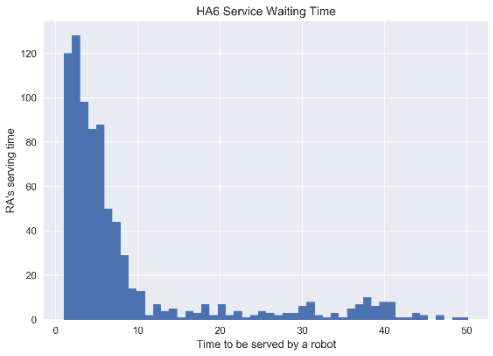

HA1 Service Waiting Time
Number of yield bags
RA's serving time

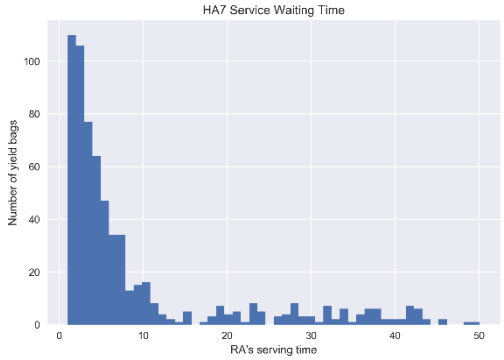

HA2 Service Waiting Time
Number of full picking bags
RA's serving time

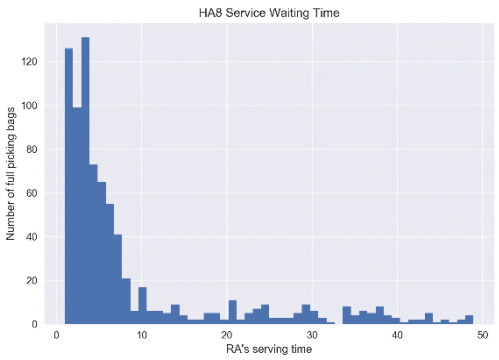

HA3 Service Waiting Time
RA's serving time
Time to be served by a robot

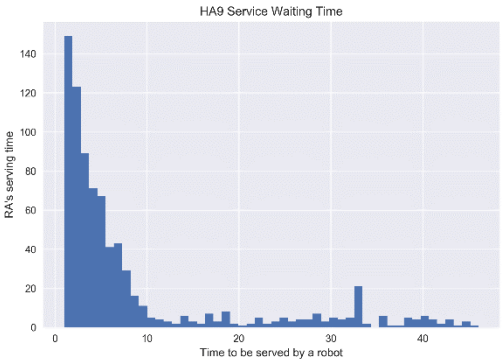

HA4 Service Waiting Time
Number of yield bags
RA's serving time

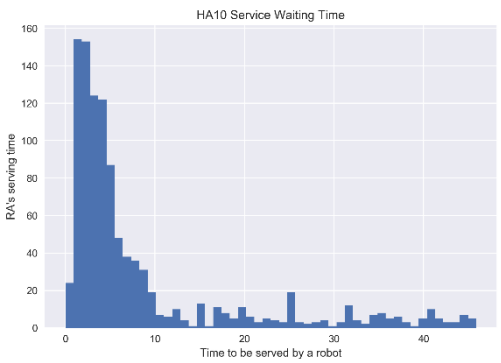

HA5 Service Waiting Time
Number of full picking bags
RA's serving time


HA6 Service Waiting Time
RA's serving time
Time to be served by a robot

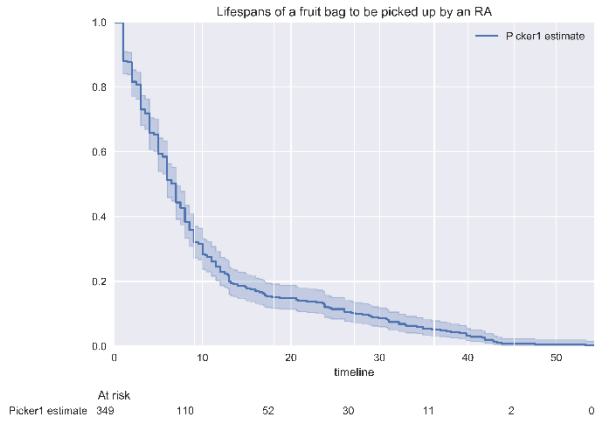

HA7 Service Waiting Time
Number of yield bags
RA's serving time

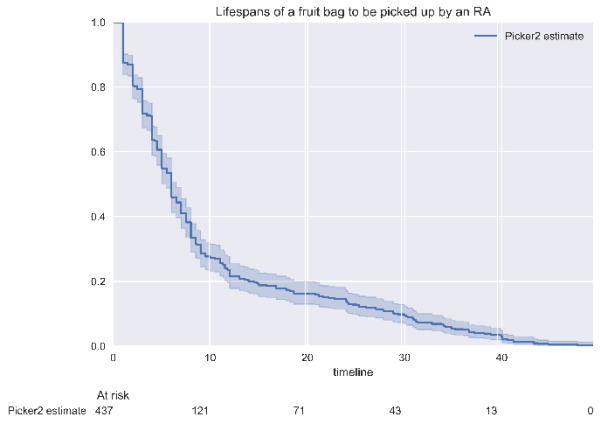

HA8 Service Waiting Time
Number of full picking bags
RA's serving time

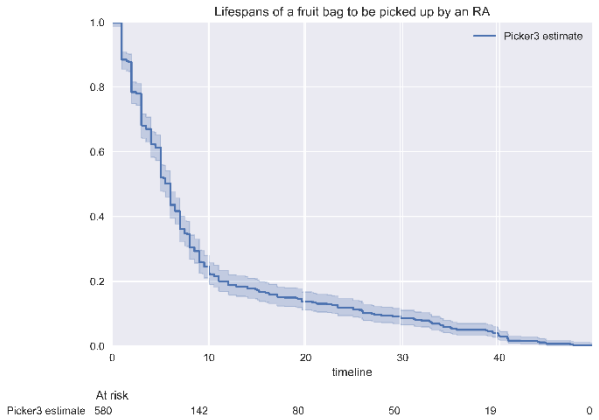

HA9 Service Waiting Time
RA's serving time
Time to be served by a robot

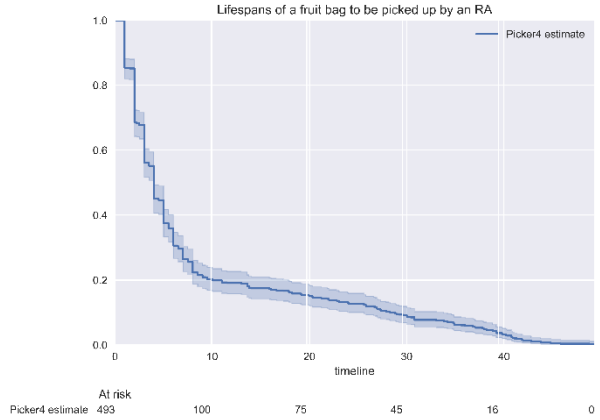

HA10 Service Waiting Time
RA's serving time
Time to be served by a robot

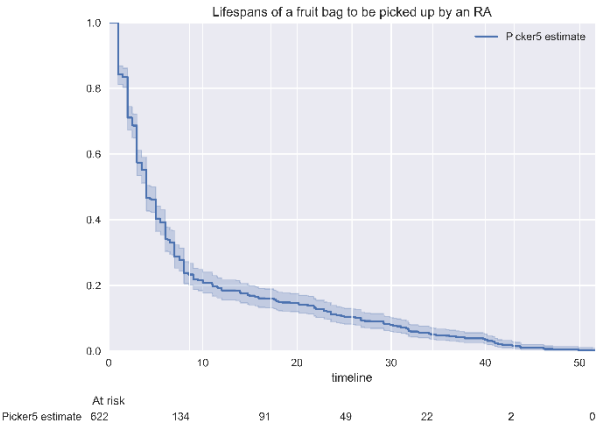
Appendix D2: FAFS algorithm simulation scenario RTA models reliability analysis serving HPA models.

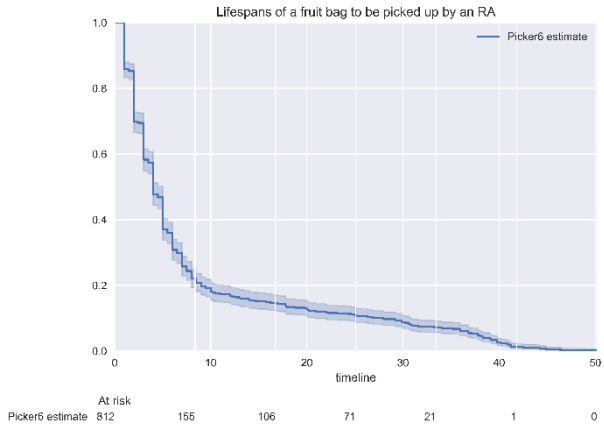


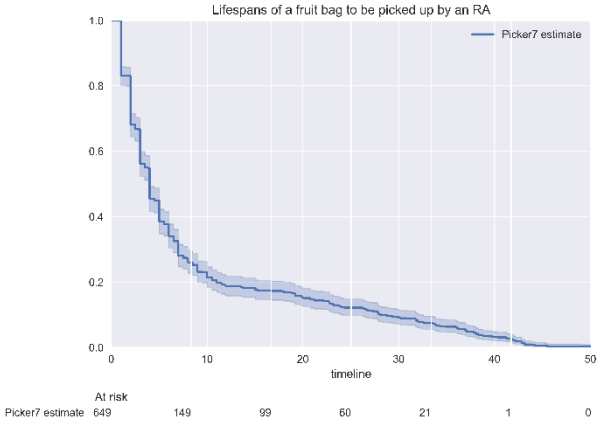


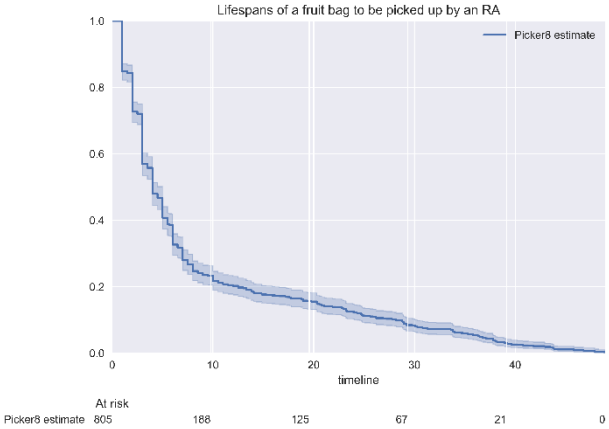




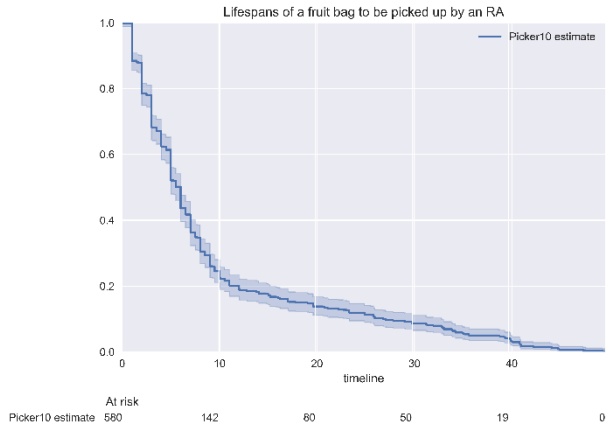




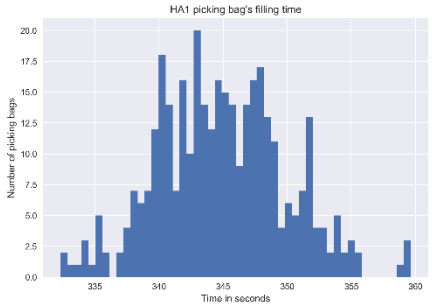

Lifespans of a fruit bag to be picked up by an RA
Picker7 estimate
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
timeline
At risk
Picker7 estimate 649 149 99 60 21 1 0

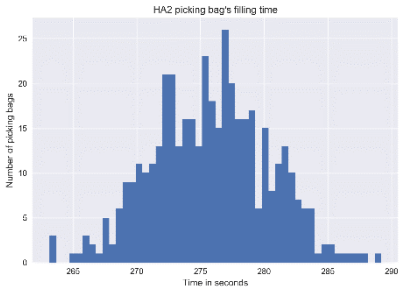

Lifespans of a fruit bag to be picked up by an RA
Picker8 estimate
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
timeline
At risk
Picker8 estimate 805 188 125 67 21 0

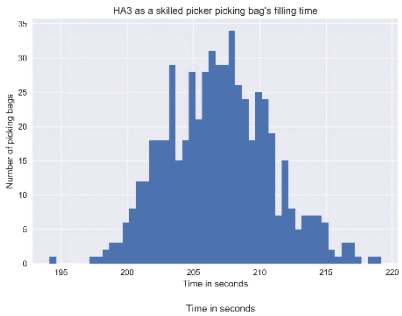

Lifespans of a fruit bag to be picked up by an RA
Picker9 estimate
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
timeline
At risk
Picker9 estimate 798 163 114 75 17 0

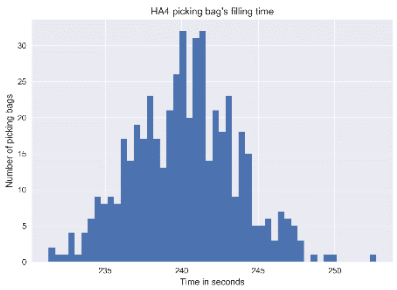

Lifespans of a fruit bag to be picked up by an RA
Picker10 estimate
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
timeline
At risk
Picker10 estimate 580 142 80 50 19 0

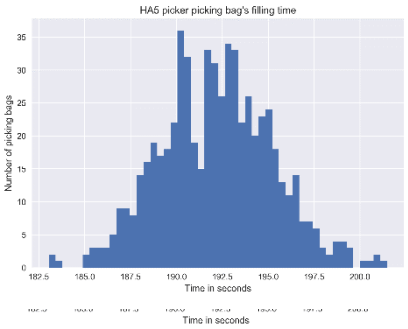
Appendix D3: DD algorithm simulation scenario HPA models request picking filling and service waiting time distribution.

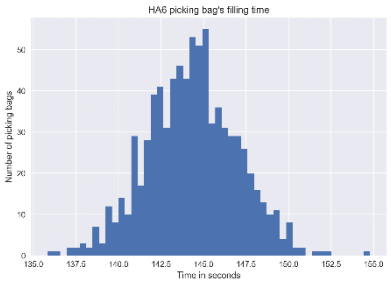


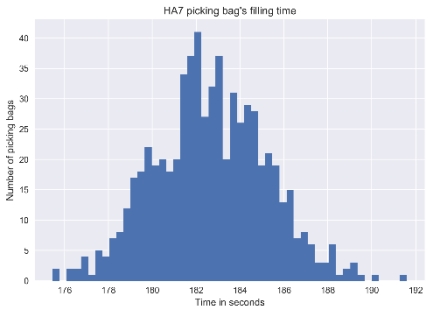


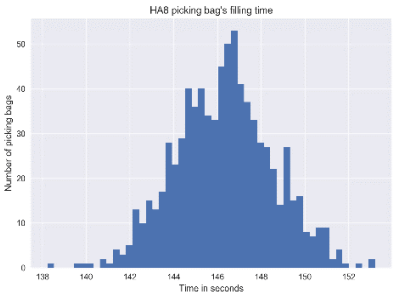




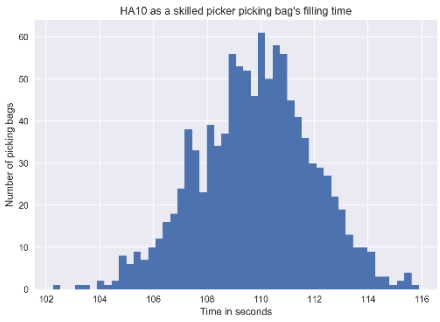


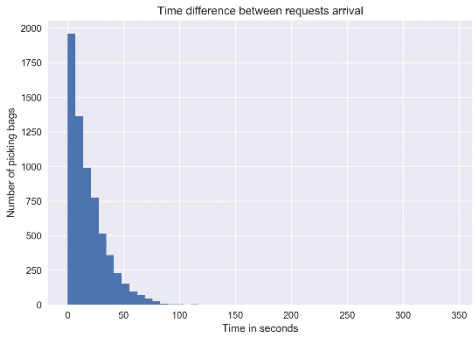


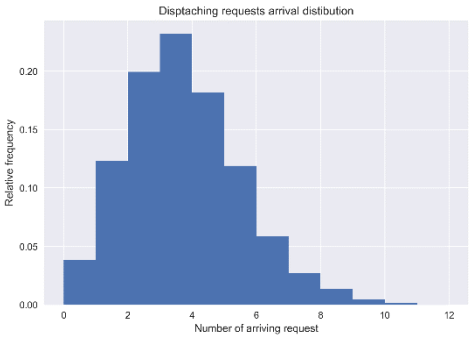


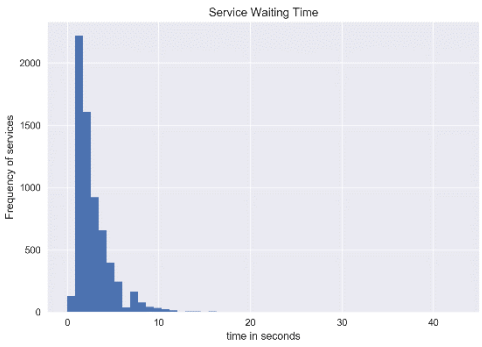


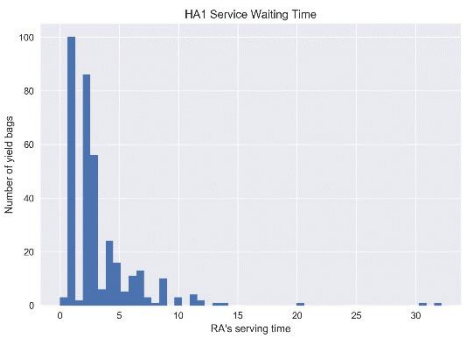

HA9 as a skilled picker picking bag's filling time
Number of picking bags
Time in seconds

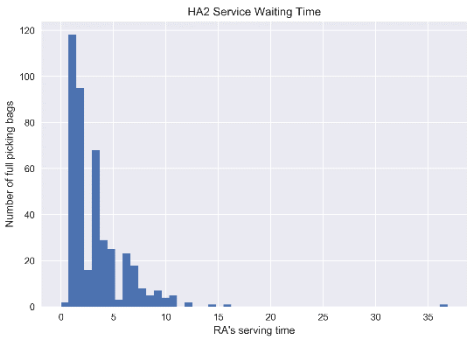

HA10 as a skilled picker picking bag's filling time
Number of picking bags
Time in seconds


Time difference between requests arrival
Number of picking bags
Time in seconds


Disptaching requests arrival distibution
Relative frequency
Number of arriving request


Service Waiting Time
Frequency of services
time in seconds

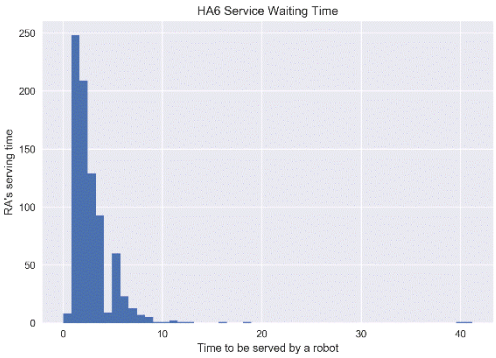

HA1 Service Waiting Time
Number of yield bags
RA's serving time


HA2 Service Waiting Time
Number of full picking bags
RA's serving time


HA3 Service Waiting Time
RA's serving time
Time to be served by a robot


HA4 Service Waiting Time
Number of yield bags
RA's serving time

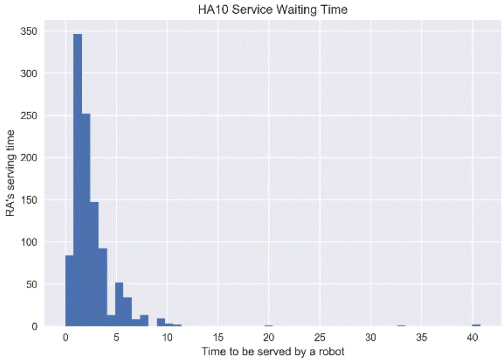

HA5 Service Waiting Time
Number of full picking bags
RA's serving time

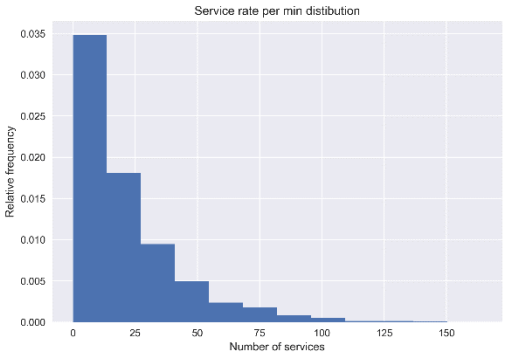

HA6 Service Waiting Time
RA's serving time
Time to be served by a robot


HA6 Service Waiting Time
RA's serving time
Time to be served by a robot

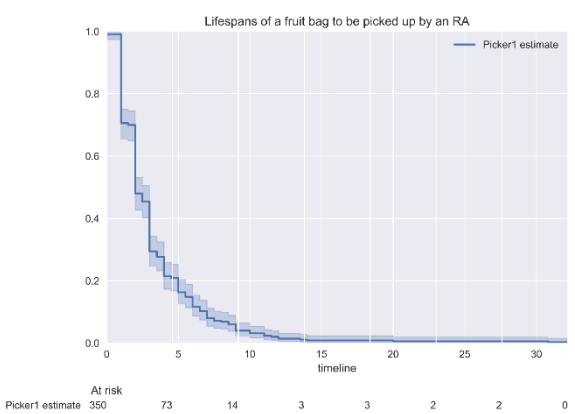

HA8 Service Waiting Time
Number of full picking bags
RA's serving time

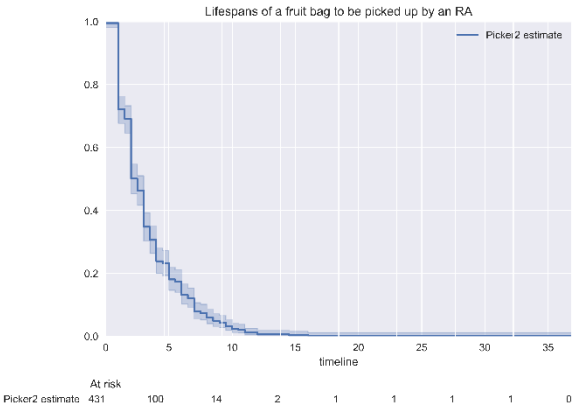

HA9 Service Waiting Time
RA's serving time
Time to be served by a robot

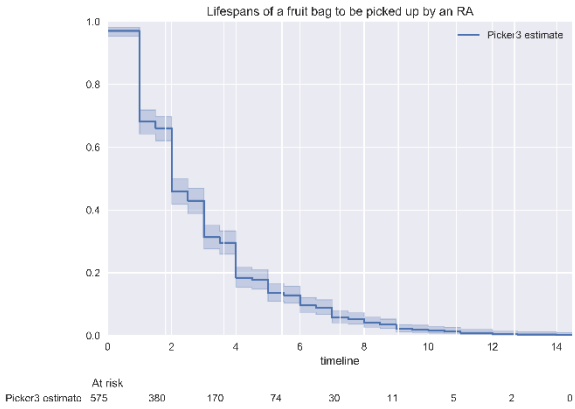

HA10 Service Waiting Time
RA's serving time
Time to be served by a robot

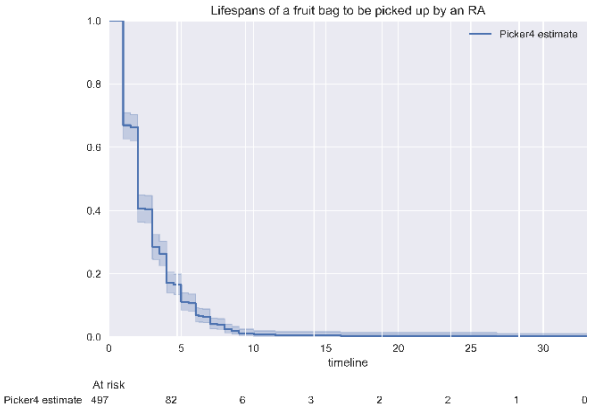

Service rate per min distibution
Relative frequency
Number of services

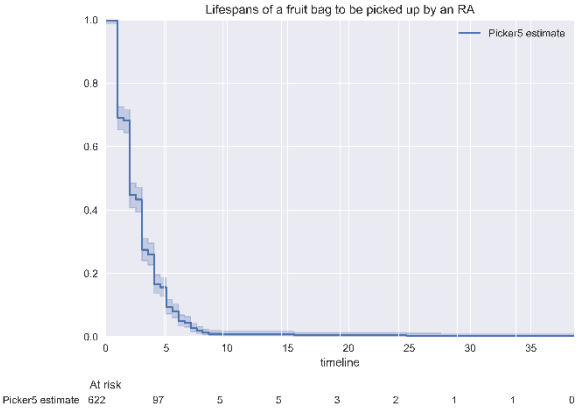
Appendix D4: DD algorithm simulation scenario RTA models reliability analysis serving HPA models.

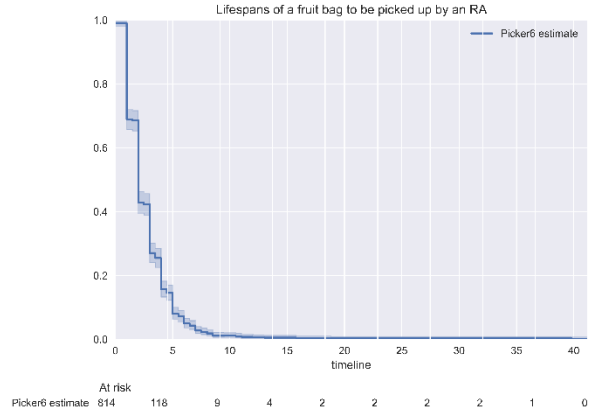


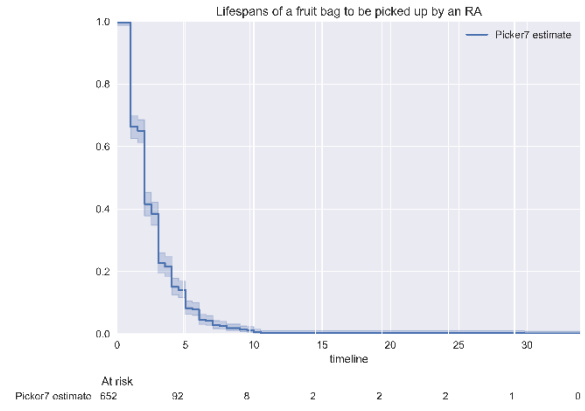


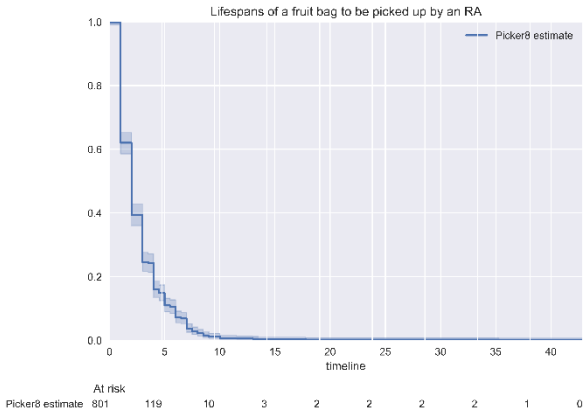


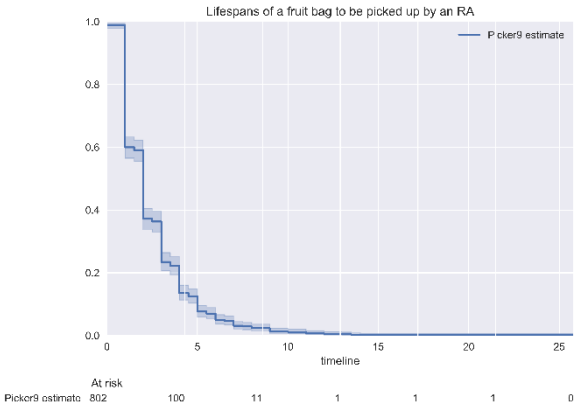


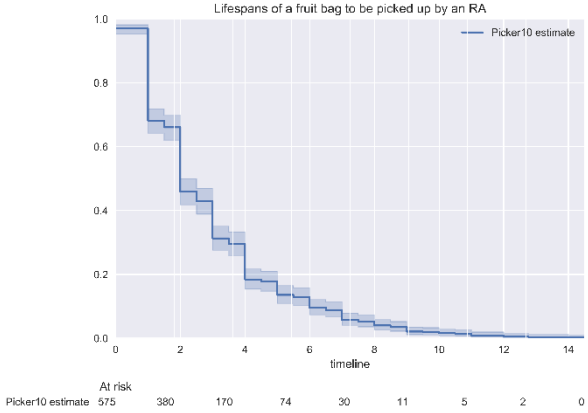


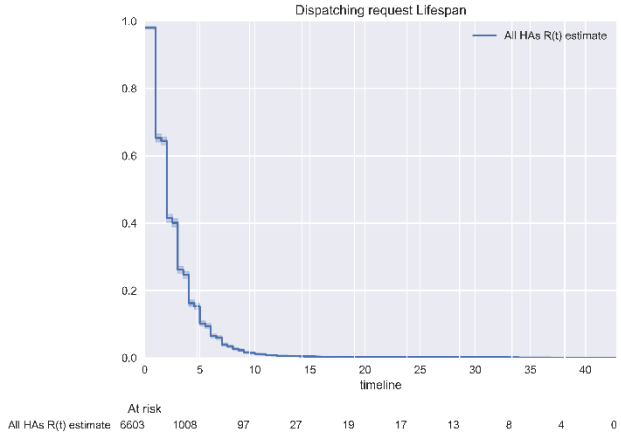


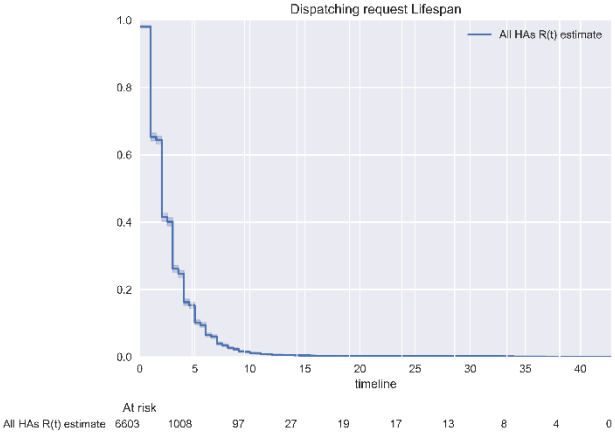

Lifespans of a fruit bag to be picked up by an RA
Picker7 estimate
timeline
At risk
Picker7 estimate 652 92 8 2 2 2 1 0


Lifespans of a fruit bag to be picked up by an RA
Picker8 estimate
timeline
At risk
Picker8 estimate 801 119 10 3 2 2 2 2 1 0

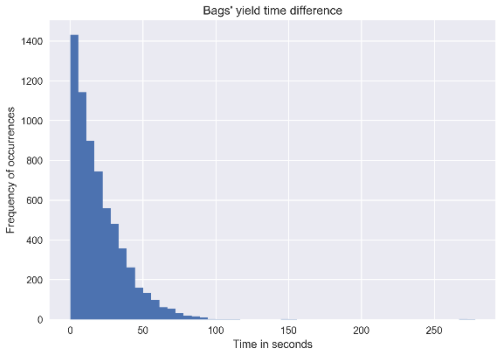

Lifespans of a fruit bag to be picked up by an RA
Picker9 estimate
timeline
At risk
Picker9 estimate 802 100 11 1 1 1 0

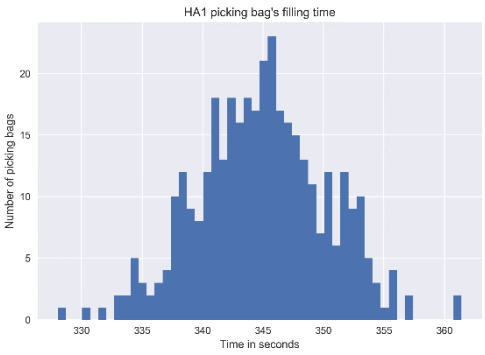

Lifespans of a fruit bag to be picked up by an RA
Picker10 estimate
timeline
At risk
Picker10 estimate 575 380 170 74 30 11 5 2 0

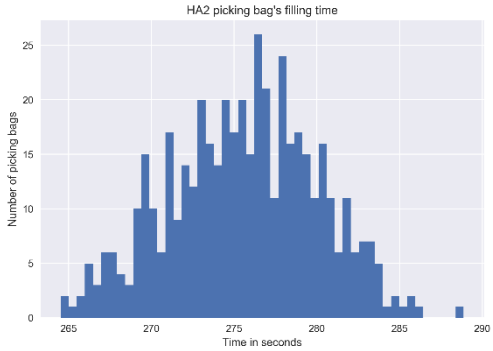

Dispatching request Lifespan
All HAs R(t) estimate
timeline
At risk
All HAs R(t) estimate 6603 1008 97 27 19 17 13 8 4 0

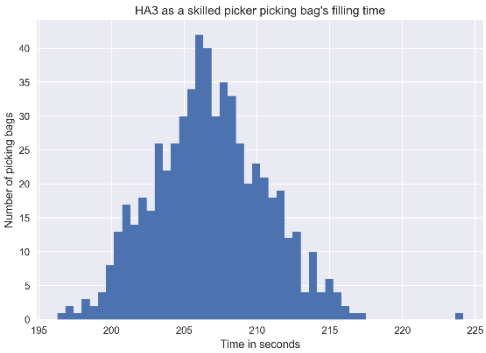

Dispatching request Lifespan
All HAs R(t) estimate
timeline
At risk
All HAs R(t) estimate 6603 1008 97 27 19 17 13 8 4 0

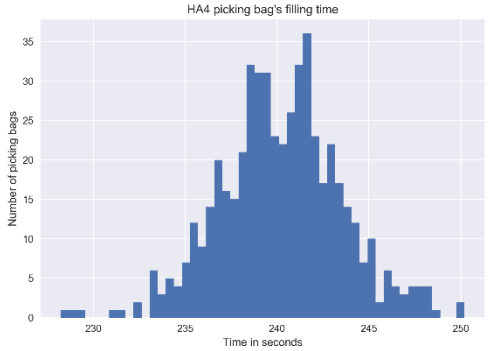
Appendix D5: DDB algorithm simulation scenario HPA models request picking filling and service waiting time distribution.

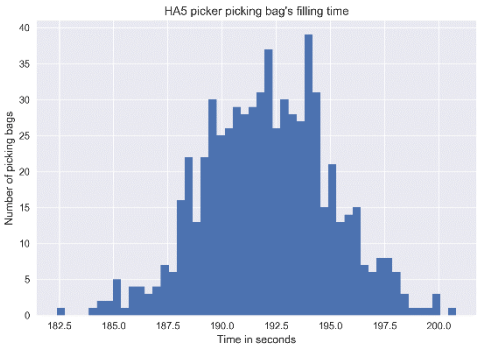


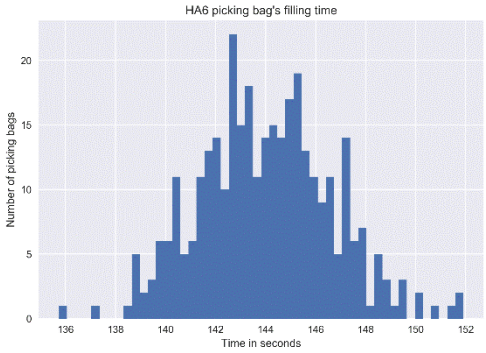


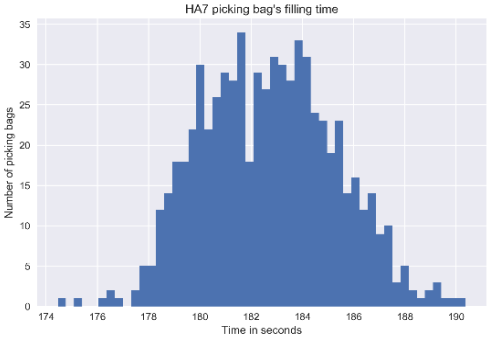




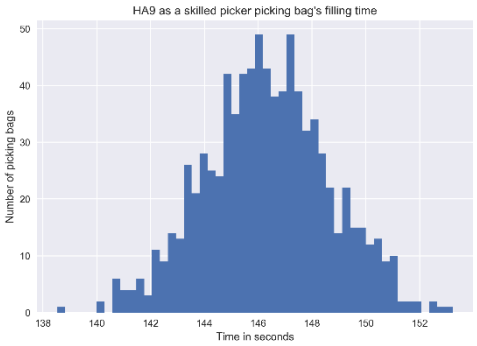


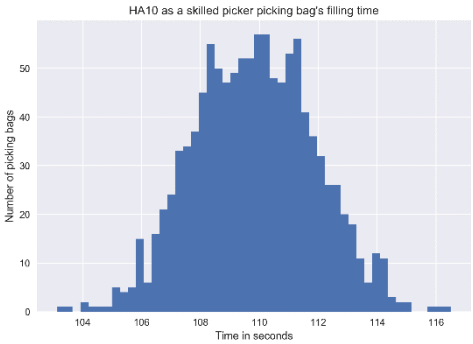


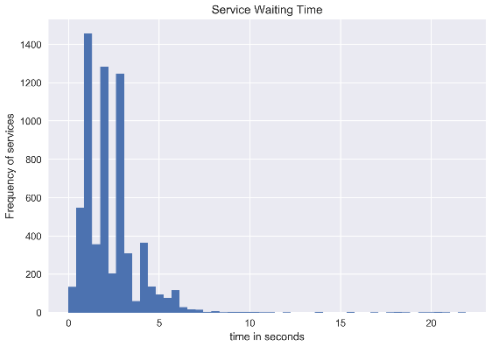


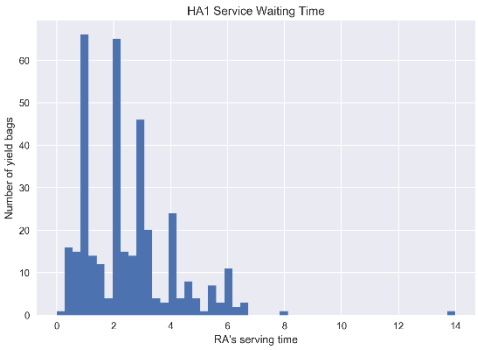



HA8 picking bag's filling time
Number of picking bags
Time in seconds

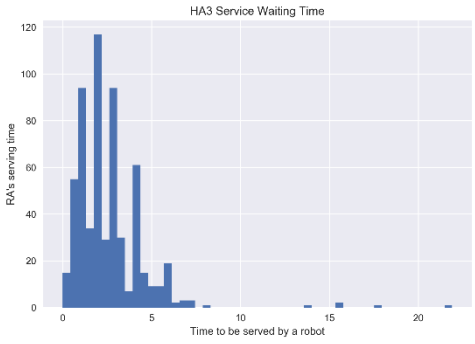

HA9 as a skilled picker picking bag's filling time
Number of picking bags
Time in seconds

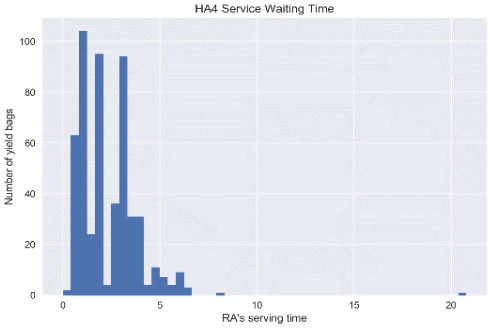

HA10 as a skilled picker picking bag's filling time
Number of picking bags
Time in seconds


Service Waiting Time
Frequency of services
time in seconds


HA1 Service Waiting Time
Number of yield bags
RA's serving time


HA2 Service Waiting Time
Number of full picking bags
RA's serving time


HA3 Service Waiting Time
RA's serving time
Time to be served by a robot


HA4 Service Waiting Time
Number of yield bags
RA's serving time

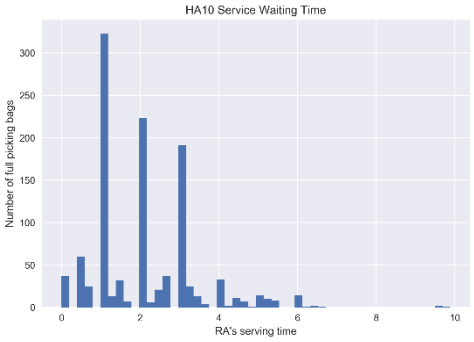

HA5 Service Waiting Time
Number of full picking bags
RA's serving time


HA6 Service Waiting Time
RA's serving time
Time to be served by a robot

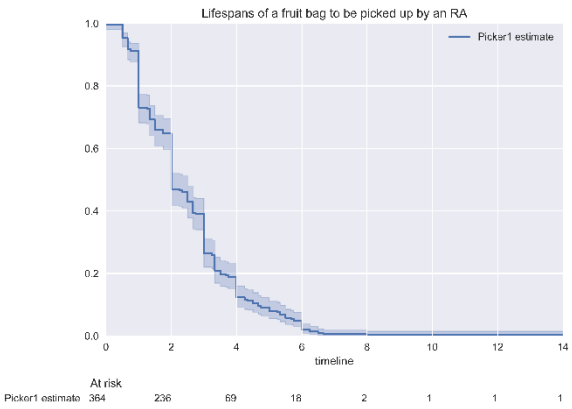

HA7 Service Waiting Time
Number of yield bags
RA's serving time

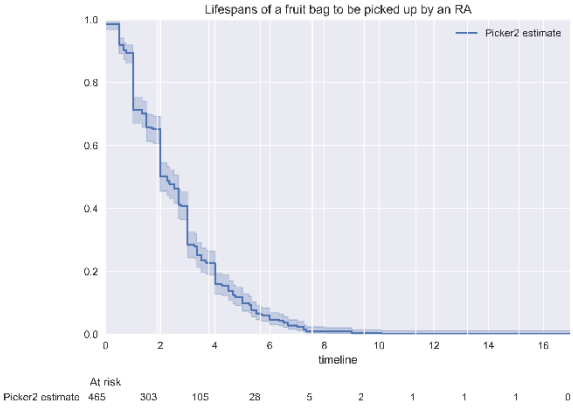

HA8 Service Waiting Time
Number of full picking bags
RA's serving time

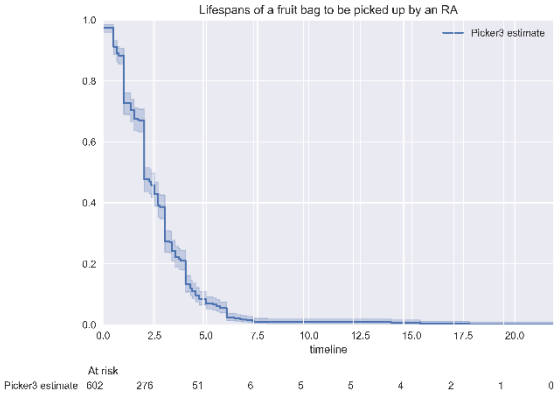

HA9 Service Waiting Time
Number of full picking bags
RA's serving time

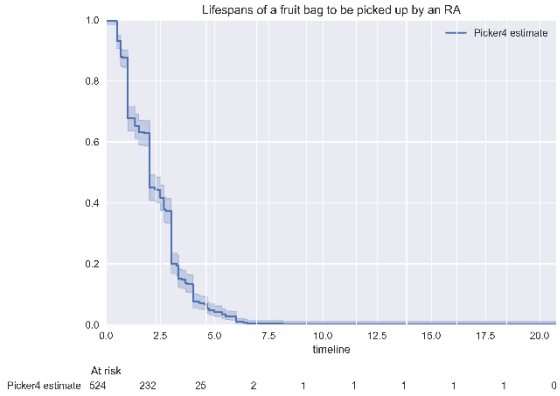

HA10 Service Waiting Time
Number of full picking bags
RA's serving time

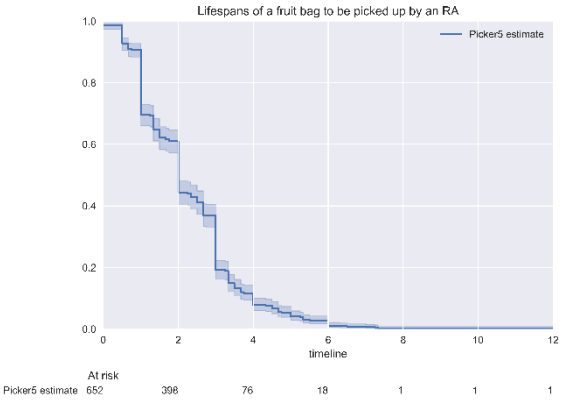
Appendix D6: DDB algorithm simulation scenario RTA models reliability analysis serving HPA models.

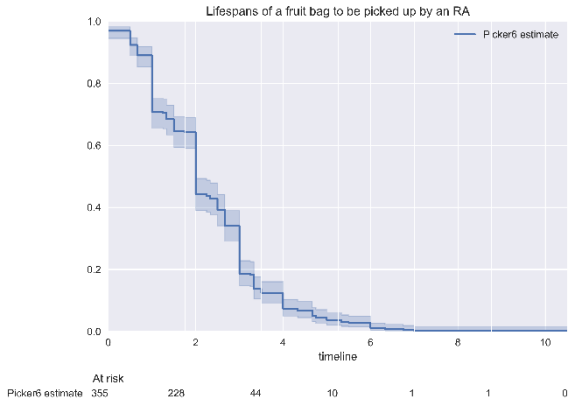


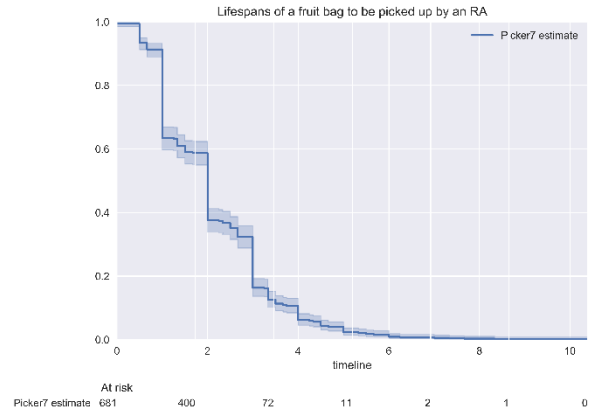


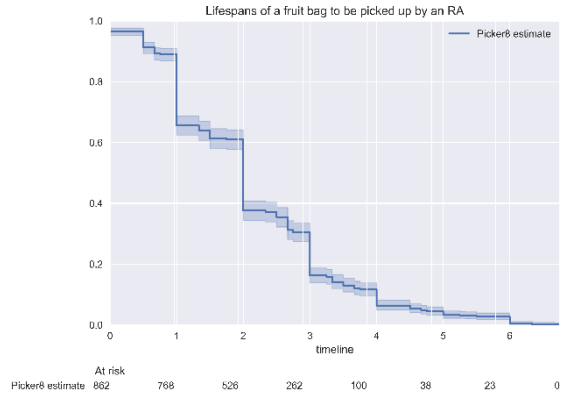


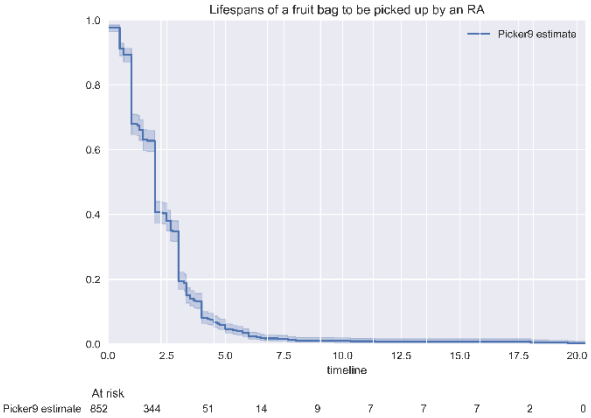


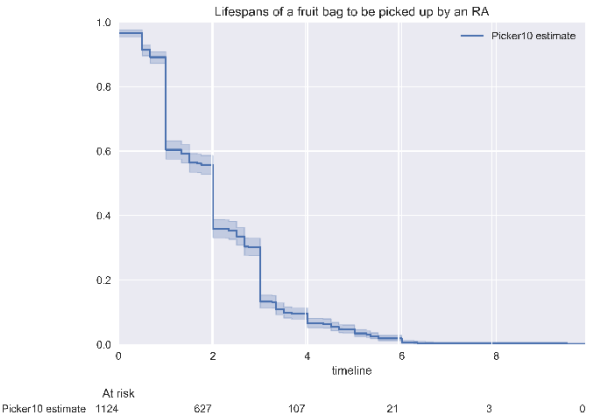


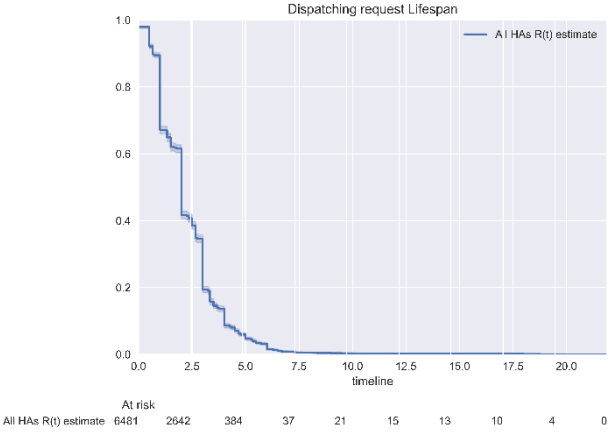



Lifespans of a fruit bag to be picked up by an RA
P cker7 estimate
timeline
At risk
Picker7 estimate 681 400 72 11 2 1 0


Lifespans of a fruit bag to be picked up by an RA
Picker8 estimate
timeline
At risk
Picker8 estimate 862 768 526 262 100 38 23 0


Lifespans of a fruit bag to be picked up by an RA
Picker9 estimate
timeline
At risk
Picker9 estimate 852 344 51 14 9 7 7 7 2 0


Lifespans of a fruit bag to be picked up by an RA
Picker10 estimate
timeline
At risk
Picker10 estimate 1124 627 107 21 3 0


Dispatching request Lifespan
All HAs R(t) estimate
timeline
At risk
All HAs R(t) estimate 6481 2642 384 37 21 15 13 10 4 0

Appendix E1: Images for training

Example image	Type of image	Number of availabl e images	Size in pixel	Explanation
	Positive image	678	64x128	The image of the HPA were manually cropped and resized by hand
	Positive	50	80x80	The image of the HPA were manually cropped, mirrored and resized by hand
	Negative images			Each image was mirrored to increase the number of samples. During training, random samples of the needed size are cropped from these images.

Appendix E2: Verification sample